НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО

Факультет інформатики та обчислювальної техніки
(назва факультету, інституту)

Кафедра автоматизованих систем обробки інформації і управління
(назва кафедри)

"На правах рукопису"
УДК *004.94;004.4;004.62*

«До захисту допущено»
Завідувач кафедри

_____ О.А.Павлов
(підпис) (ініціали, прізвище)

“____” _____ 20*18* р.

МАГІСТЕРСЬКА ДИСЕРТАЦІЯ

на здобуття ступеня магістра

за спеціальністю *122 Комп'ютерні науки та інформаційні технології*
(код та назва спеціальності)

спеціалізацією *Інформаційні управляючі системи та технології*
(код та назва спеціалізації)

на тему: *Прогнозування потреби в ресурсах для серверної системи в умовах хмарних обчислень*

Виконав: студент *VI* курсу групи ІС-61м
(шифр групи)

Терентьєв Роман Анатолійович _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник *доц., к.т.н., доц. Жаріков Е.В.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант *к.т.н., доц. Жданова О.Г.* _____
(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2018

РЕФЕРАТ

Магістерська дисертація: 95 с., 27 рис., 2 табл., 1 додаток, 50 джерела.

Актуальність. У сучасному світі обсяг інформації щорічно подвоюється, при цьому збільшується швидкість ведення бізнесу. Щоб бути успішною, сучасноій компанії доводиться не просто оперувати великими обсягами даних, а оперувати ними швидко та ефективно.

Центр обробки даних [1] — це відмовостійка комплексна централізована система, що забезпечує автоматизацію бізнес-процесів з високим рівнем продуктивності та якістю сервісів. Звертаючись до великомасштабних інформаційних завдань, більшість компаній стикаються з неконтрольованими зростаючими витратами та зниженням ефективності інвестицій в IT. Створення дата-центру та консолідація IT-ресурсів забезпечують якісний підйом у розвитку корпоративних систем, відкриваючи нові можливості для впровадження найсучасніших технологій.

Конструкція центру обробки даних [2] здійснюється з урахуванням критичних бізнес-задач, рівень вимог безпеки, використання існуючого обладнання, і втілений в архітектурних і технічних рішень проекту. Цей підхід дозволяє створювати захищені гетерогенні центри обробки даних, які складаються з апаратного та програмного забезпечення різних виробників, у тому числі успадкованих систем. Забезпечуючи постійне дотримання вимог користувачів із мінімізацією витрат, існує проблема прогнозування ресурсів серверних ресурсів у хмарних обчисленнях.

У зв’язку з цим актуально є розробка алгоритму прогнозування потреби ресурсів серверної системи в умовах хмарних обчислень, який дозволить більш точно прогнозувати завантаженість системи.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалась на кафедрі автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» в рамках теми «Розробка та впровадження системи

управління IT-інфраструктурою з консолідованими інформаційно-обчислювальними ресурсами» (№ 0115U000322).

Метою дослідження є поліпшення якості прогнозування потреби ресурсів для серверної системи шляхом розробки алгоритму прогнозування для збільшення точність прогнозування завантаженості серверної системи.

Для досягнення поставленої мети мають бути виконані наступні завдання:

- проаналізувати предметне прогнозування ресурсів для серверної системи;
- провести огляд методів прогнозування споживання ресурсів серверної системи;
- розробити алгоритм для прогнозування потреби ресурсів серверної системи;
- виконати програмну реалізацію алгоритму;
- провести дослідження ефективності розробленого алгоритму.

Об’єктом дослідження є процес прогнозування потреби ресурсів для серверної системи в умовах хмарного обчислення.

Предметом дослідження є методи і алгоритми прогнозування потреби ресурсів для серверної системи в умовах хмарного обчислення.

Методами дослідження є методи прогнозування, які базуються на авторегресійних моделях та нейронних мережах.

Наукова новизна отриманих результатів. Проаналізовано можливість застосування методів прогнозування для передбачення завантаженості ресурсів для серверної системи в умовах хмарних обчилсень. Розроблено гібрідний метод прогнозування, на основі використання авторегресійних моделей та нейронних мереж. Розроблений алгоритм враховує зміни навантаження на центральний процесор сервера та точність прогнозування методів на попередньому кроці прогнозування з метою підвищення точності наступного кроку прогнозування. Запропонований алгоритм базується на моделях авторегресії, авторегресії ковзного середнього, інтегрованій авторегресії ковзного середнього та на методі групового урахування аргументів. Він дозволяє визначати наближену до оптимальної політику

управління режимами роботи фізичного сервера без попередньої інформації про навантаження.

Публікації. Матеріали роботи опубліковані у тезах 10-ї Всеукраїнської науково-практичної конференції «Комп'ютерні інтелектуальні системи та мережі» [3]; опубліковані у тезах 18-ї Всеукраїнської студентської науково-практичної конференції «Наука та техніка XXI століття» [4]; опубліковані у тезах науково-практичної конференції «Інформатика та обчислювальна техніка-IOT-2018» [5]; опубліковані в журналі «Наукові вісті Далівського університету» [6]; опубліковані в журналі «Актуальные научные исследованияв современном мире» [7].

ПРОГНОЗУВАННЯ, АВТОРЕГРЕСІЙНІ МОДЕЛІ, МЕТОД ГРУПОВОГО УРАХУВАННЯ АРГУМЕНТІВ, СЕРВЕРНІ СИСТЕМИ, CLOUD COMPUTING, ЦЕНТР ОБРОБКИ ІНФОРМАЦІЇ

ABSTRACT

Master dissertation: 95 pp., 27 fig., 2 tab., 1 app., 50 sources.

Topicality. In the modern world, the volume of information is doubling annually, and at the same time the speed of doing business is increasing. To be successful, a modern company has to not just operate large volumes of data, but operate them quickly and efficiently.

Data center (data center) is a fault-tolerant integrated centralized system that provides automation of business processes with high level of productivity and quality of provided services. Addressing large-scale information tasks, most companies face uncontrolled rising costs and lower efficiency in IT investments. The construction of data centers and the consolidation of IT resources provides a qualitative leap in the development of corporate systems, opening new opportunities for the implementation of the most advanced efficient technologies.

The design of the data center is performed taking into account the solvable business tasks, the level of security requirements, the use of existing equipment, and is embodied in the architectural and technical decisions of the project. This approach allows for the creation of protected heterogeneous data processing centers, consisting of hardware and software from various manufacturers, including legacy customer systems. In order to ensure that the requirements of users are constantly adhered to minimizing costs, it is a problem of forecasting the server system resource requirements in cloud computing.

In this regard, it is important to develop an algorithm for forecasting the demand for resources of the server system in a cloud computing environment, which will more accurately predict the system's load.

Relationship of work with scientific programs, plans, themes. The research was carried out at the Department of Computer-Aided Management And Data Processing Systems of the National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute» within the theme «Development and implementation of an IT infrastructure

management system with consolidated information and computing resources» (№ 0115U000322).

The aim of the research is to improve the quality of prediction of resource requirements for the server system by developing a prediction algorithm to increase the accuracy of forecasting the load of the server system.

To achieve the goal, the following tasks must be performed:

- analyze the methods of predict resources of the server system
- develop an algorithm for predict the server system resource requirements;
- develop the software implementation of the algorithm;
- develop the software implementation of the reinforcement learning algorithm;
- make a research of developed algorithm effectiveness.

The object of research is a process of predicting resource requirements for a server system in a cloud computing environment.

The subject of research are predicting methods and algorithms necessary resources for server system in a cloud computing.

Research methods are predicting methods based on autoregressive models and neural networks.

Scientific novelty of the obtained results. The possibility of using forecasting methods to predict the loading of resources for the server system in the conditions of cloud computing is analyzed. The mixed forecasting method is developed, based on autoregressive models and neural networks. The developed algorithm takes into account changes in the load on the central processing unit of the CPU and the accuracy of forecasting of the methods in the previous step in order to increase the accuracy of the next steps of forecasting. The proposed algorithm is based on autoregressive models, autoregression of the moving average, integrated automorregression of the moving average and the method of group consideration of arguments allows to determine the optimal policy of controlling the modes of operation of the physical server without prior information on the load.

Publications. The materials of research are published in the theses of the 10th All-Ukrainian Scientific and Practical Conference "Computer Intelligent Systems and Networks" [3]; published in the theses of the 18th All-Ukrainian Student Scientific and Practical Conference "Science and Technology of the XXI Century" [4]; published in the theses of the scientific-practical conference "Informatics and Computing Technology-IOT-2018" [5]; published in the journal "Scientific News of Daliv University" [6]; published in the journal "Actual research in the modern world" [7].

PREDICTING, AUTHORIZATION MODELS, GROUP CALCULATION METHOD OF ARGUMENTS, SERVER SYSTEMS, CLOUD COMPUTING, INFORMATION PROCESS CENTER

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

AR – англ. Autoregressive – авторегресія.

MA – англ. Moving average – ковзне середнє.

ARMA – англ. Autoregressive moving average – авторегресія ковзного середнього.

ARIMA – англ. Autoregressive integrated moving average – інтегрована модель авторегресії ковзного середнього.

GMDH – англ. Group method of data handling – метод групового врахування аргументів.

MAPE – англ. Mean absolute percentage error – Середнє абсолютнє відхилення

RMSE – англ. Root-mean-square deviation – середньоквадратичне відхилення.

SLA – англ. Service-level agreement – угода про рівень послуг.

ЦОД – центр обробки даних.

ВМ – віртуальна машина.

CPU – англ. Central processor unit – центральний процесор.

ПЗ – програмне забезпечення.

ФС – фізичний сервер.

СОА – сервіс-орієнтована архітектура.

ITSM – англ. IT Service Management – керування IT-послугами.

LR – англ. Local Regression – локальна регресія.

QoS – англ. Quality of Service – якість обслуговування.

SVM – англ. Vector Machine Support - метод опоних векторів.

NN – англ. neural networks – нейронна мережа.

ВСТУП

Швидке зростання попиту [8] на обчислювальну потужність сучасних сервісних додатків в поєднанні з переходом до моделі хмарних обчислень на основі центрів обробки даних (далі за текстом — ЦОД) привели до необхідності дослідження процесів, що відбуваються у віртуалізованому середовищі, та розробки і впровадження систем управління ресурсами ЦОД. Сучасні ЦОД споживають величезну кількість електроенергії, що призводить до високих експлуатаційних витрат.

Модель хмарних обчислень виявилася дуже корисною і набула широкої популярності, її використовують тисячі компаній та корпорацій, що є провайдерами послуг в сфері ІКТ. Хмарні ЦОД використовуються для різних рішень, таких як обчислення ресурсномістких задач, побудова віртуальних приватних мереж, обробка великих обсягів даних, збереження даних, але переважно вони використовуються для різних мережевих застосунків та сервісів великих корпорацій. Виявилося, що ефективна організація процесів управління ресурсами ЦОД пов’язана з необхідністю розв’язання низки проблем, насамперед створення умов для функціювання інформаційно-обчислювальних та комунікаційних потужностей ЦОД, управління віртуалізованими ресурсами, забезпечення надійності та безпеки. Схожі проблеми постають перед корпораціями з розвиненою розподіленою IT-інфраструктурою, насамперед хостинговими компаніями. Вкладаючи кошти, компанії сподіваються на прибуток. У будь-якому випадку вони очікують зменшення витрат на експлуатацію ЦОД, зниження вартості обслуговування користувачів, що дозволить, зрештою, закласти основу для ефективної діяльності як самої компанії, так і клієнтів.

Клієнти своє бачення роботи IT-інфраструктури погоджують із хостинговою компанією на рівні вимог SLA (Service-level agreement), до яких звичайно належать: вартість послуг, доступність і керованість IT-інфраструктури, цілісність даних, безпека, надійність, масштабованість.

Досягнення рівня вимог користувачів найменшими коштами становить сутність проблеми створення систем управління і забезпечення функціонування ЦОД. Однією з них є проблема прогнозування навантаження ЦОД для того, щоб можна було заздалегідь мігрувати віртуальні машини на інші фізичні сервери, щоб вивільнити зайву потужність всієї серверної системи або навпаки збільшити, для забезпечення оперативності при реагуванні на збільшення запитів користувачів та якісної роботи хмарних обчислень з найменшими витратами.

Необхідно розробити ефективні рішення, які ґрунтуються на зборі інформації за попередній період та її обробки про стан ресурсів та обсяги навантаження на ЦОД. Для постійної підтримки ефективного функціювання ЦОД потрібно робити середньострокові прогнози та інколи довгострокові прогнози, та передавати дані в систему управління ресурсами для міграції ВМ на інші ФС. Їх створення становить важливу науково-практичну проблему, розв’язання якої вимагає розуміння процесів, які відбуваються в хостингових компаніях, процесів функціювання ІТ-інфраструктури, чіткої постановки конкретних завдань дослідження, розробки математичних моделей підсистем ЦОД і відповідних методів вирішення задачі та реалізації згаданих вище методик і алгоритмів.

Комплекс задач дослідження та їх постановки залежить від багатьох чинників, які тією чи іншою мірою впливають на згадані вище процеси. Першим таким чинником є модель хостингу. Другим важливим чинником є прийнята архітектура побудови програмних систем. Це традиційна клієнт-серверна архітектура, або сервіс-орієнтована архітектура (СОА). Перша з них ґрунтується на абстракціях застосувань та елементів їх побудови. Переважно використовується трирівнева архітектура із сервером БД, сервером застосувань і клієнтом. У СОА ідеологія побудови виходить із абстракції сервісів. Мета розроблення системи прогнозування полягає у створенні комплексу прийнятного рівня абстракцій сервісів, які використовуються багатьма застосуваннями, а не жорстко прив’язуються до одного з них. Архітектура СОА набула подальшого розвитку в умовах хмарних обчислень. Для досягнення поставленої мети мають бути виконані наступні завдання:

- проаналізувати предметне прогнозування ресурсів для серверної системи;
- провести огляд методів прогнозування ресурсів серверної системи;
- розробити алгоритм для прогнозування потреби ресурсів серверної системи;
- виконати програмну реалізацію алгоритму;
- провести дослідження ефективності розробленого алгоритму.

Об'єктом дослідження є процес .

1 ОГЛЯД РІШЕНЬ З ПРОГНОЗУВАННЯ РЕСУРСІВ ДЛЯ СЕРВЕРНОЇ СИСТЕМИ В УМОВАХ ХМАРНИХ ОБЧИСЛЕНЬ

У статті [9] розглядаються економічні аспекти використання технологій віртуалізації в IT-середовищах підприємств.

Постановка проблеми. Для ефективного ведення бізнесу і забезпечення конкурентоспроможності як малі, так і великі компанії потребують інновацій. Причому, впровадження нових технічних рішень може виявитися досить дорогим процесом, зважаючи на складність управління IT інфраструктурою, що розширюється. Витрати на інновації для зростаючого бізнесу, зважаючи на обмежений бюджет, можуть виявитися неприйнятними. IT інфраструктура, побудована з розрахунку на вчорашні запити бізнесу, яка не підтримує зростаючі потреби в нових функціях і сервісах, залишається складною і негнучкою, а тому і дорогою в підтриманні і обслуговуванні. Одним із варіантів рішення цих проблем є перехід до віртуалізації обчислень на базі ЦОД.

У той же час віртуалізація зберігання даних дозволяє об’єднати ресурси декількох систем зберігання в єдиний пул і управляти ними як єдиним фізичним пристроєм. Але окрім фізичних пристроїв, сучасні технології віртуалізації охоплюють і нематеріальні сутності: робоче середовище користувача, простори імен, зони безпеки та ін. Таким чином, з проведеного вище аналізу ми можемо визначити, що технології віртуалізації — це програмні, апаратні або апаратно-програмні технології, що дозволяють подати матеріальні і нематеріальні ресурси у вигляді одного або декількох абстрактних ресурсів з різними цілями: розподіл на незалежні середовища, логічне об’єднання, емуляція іншого ресурсу, зручність виконання операцій з експлуатації.

Також очевидно, що, зважаючи на поширення технологій віртуалізації, виникає питання управління ресурсами в умовах віртуальних середовищ. Ми вважаємо, що це приведе до істотного розвитку засобів моніторингу, прогнозування навантаження на ФС, діагностики й управління віртуалізацією і, також, активно

розвиватиметься наступний функціонал засобів віртуалізації: простота використання, управління додатками, висока доступність, оптимізація ресурсів, безперервність надання сервісів, забезпечення роботи мобільних користувачів, відновлення при аварійних ситуаціях, балансування навантаження та ін.

У роботі [10] розглянутий один з методів побудови сучасної IT-інфраструктури для забезпечення гнучкого розподілення різноманітних ресурсів корпоративних мереж та ефективного їх використання. Підхід, що пропонується, має тенденцію розвитку убік сервіс-орієнтованої IT-інфраструктури. вимагає від IT гнучкості, високої якості обслуговування і ефективності, а ці запити можна задовольнити за допомогою сервіс-орієнтованої архітектури і сервіс-орієнтованої IT-інфраструктури (далі за текстом - COI). Перша дозволяє об'єднати сервіси, що надаються застосуваннями, для підтримки учбово-виробничих процесів, а друга – динамічно виділяти додаткам ресурси. Основними характеристиками COI є: автоматичний розподіл ресурсів для додатків; інтегроване управління і моніторинг ресурсів; стандартизація, що забезпечує сумісність, і автоматизація IT-операцій. Стратегія COI втілена в концепції динамічного центру даних (DDC). Її реалізація передбачає віртуалізацію ресурсів, їх динамічний розподіл і інтеграцію стандартних рішень і IT-сервісів. Для динамічних IT-сервісів пропонується створення віртуального серверного пулу, в якому ресурси організовані за допомогою спеціальної архітектури. Вона наділяє це рішення всіма атрибутами SOA для використання в центрах обробки даних нового покоління. Максимальна утилізація продуктивності, що надається новими процесорами, краще всього досягається за допомогою віртуалізації. Віртуалізація є ефективною, якщо її підтримують всі системи. Такий підхід гарантує, що у віртуальному середовищі додатки працюватимуть з високою швидкодією. Віртуалізація в будь-якій своїй реалізації належить до найбільш перспективних і вигідних рішень в області IT індустрії. Технологія віртуалізації, що дозволяє абстрагувати один від одного різні компоненти IT-систем, давно використовується для консолідації серверних ресурсів, в системах зберігання даних, а також для створення клієнтських робочих

місць, що дозволяють забезпечити користувачеві доступ до робочих матеріалів з будь-якого терміналу, включаючи територіально віддалені. Окрім консолідації, серверна віртуалізація може застосовуватися для здійснення роботи застарілих застосувань, здатних функціонувати тільки як «гостьові» на сучасному устаткуванні під управлінням більш застарілих ОС. Крім того, існує проблема балансування навантаження між всіма фізичними серверами, що існують в організації. Як правило, здійснити таке балансування раціональним чином украй важко, і, як наслідок, частина серверів виявляється переобтяженою, тоді як існує велика кількість мало-завантажених серверів, потужності яких використовуються на 5-10%.

Для вирішення проблем прогнозування навантаження серверів, потрібно спробувати реалізувати та порівняти такі методи як:

- авторегрессивна інтегрована змінна середня (autoregressive integrated moving average, далі за текстом — ARIMA); серверної системи;
- Баєсове прогнозування;
- адаптивна змінна середня Кауфмана.

У публікації [11] на основі узагальнення накопиченого досвіду розв’язання проблеми розподілу і управління навантаженням і ресурсами центрів оброблення даних пропонують модифікації раніше розроблених моделей, які більш вдало враховують реалії цієї актуальної галузі досліджень. З метою розроблення універсального алгоритму для цього класу задач авторами пропонується варіант керованого ГА, що полягає у використанні механізма отримання нової популяції особин на базі представників минулої епохи. Керованість означає вплив на вибір чергових операторів формування популяції у залежності від деяких важливих її параметрів. Шляхом налаштування зазначених параметрів керований ГА можна досить швидко навчити розв’язувати різноманітні проблеми зазначеного класу, особливістю яких є складний характер взаємодії критеріїв і обмежень, що раніше часто призводило до передчасного виродження популяції.

Експериментальне дослідження показало, що керований ГА дозволяє отримати покращені результати відносно базового ГА. У більшості випадків

керований варіант отримував розв’язки швидше за кількістю епох за базовий. Налаштовуючи систему правил керованого ГА, можна для кожної задачі підібрати схему значень, за якою за орієнтовну кількість кроків (епох) можна досягти розв’язку достатнього рівня оптимальності. Однак, недоліком цього алгоритму є те що пошук таких схем параметрів вимагає витрат часу та експериментальної роботи. Керований ГА покладений в основу реалізації функцій оптимізації створеної в НТУУ «КПІ» СОІ SmartBase.ITSControl.

У публікації [12] розглянуто математичні моделі та побудову функцій прогнозування на основі цих моделей для таких характерних класів економічних процесів: стаціонарних процесів авторегресії та авторегресії з ковзним середнім, нестаціонарних процесів із детермінованими та стохастичними трендами, гетероскедастичних та коінтегрованих процесів. Наведено структури математичних моделей та приклади їх застосування для короткострокового та середньострокового прогнозування. При цьому довгостроковий прогноз визначається як безумовне математичне сподівання розв'язку різницевого рівняння, а короткостроковий і довгостроковий — через умовне математичне сподівання.

Показано, що для опису випадкового тренду можна застосовувати модель випадкового кроку з дрейфом та шумом, а у складніших випадках необхідно скористатись моделлю лінійного локального тренду. Для моделювання гетероскедастичних процесів існує досить широкий вибір структур рівнянь, які дають можливість прогнозувати дисперсію процесу з досить високою точністю. Для класу коінтегрованих процесів також існує напрацьована методика їх побудови, яку можна застосовувати у випадку двох та більше змінних. Однією із основних проблем при моделюванні часових рядів залишається створення процедури правильного вибору класу та структури рівнянь для їх опису. Для прискорення цієї процедури необхідні системи підтримки прийняття рішень при моделюванні та прогнозуванні динаміки часових рядів.

У публікації [13] розглядається прогнозування продуктивності комп'ютерної мережі за допомогою методу ARIMA. Прогнозування мережевого трафіку становить

значний інтерес в таких областях як відстеження перевантажень в мережі, контроль потоків даних та мережеве управління. Ретельно підібрана модель трафіку здатна виявити і передбачити найважливіші характеристики мережевого трафіку: короткочасно і довготривало залежні процеси, самоподобна на великих часових масштабах. У даній статті представлена модель ARIMA з мінімальним числом параметрів, що має адекватний прогноз. Запропонована процедура оцінки параметрів моделі ARIMA і вибору моделі з мінімальним числом параметрів. Наведено порівняння оцінок якості прогнозу для отриманих моделей.

Побудова систем on-line моніторингу з можливістю динамічного управління виділеною пропускною спроможністю лінії зв'язку для певних видів сервісу - важлива перспективне завдання. Широке поширення сервісів розподіленого зберігання даних P2P, відсутність «вузького горла» як елемента системи, що обмежує загальний обсяг трансльованих даних, дозволяють розглядати подальший розвиток мереж P2P як фактор, що вносить дисбаланс в існуючу мережеву інфраструктуру. Впровадження систем динамічного управління здатні збалансувати навантаження, що виникає в різні часові періоди, і виключити виникнення перевантажень в сегментах мережі без істотного впливу на якість роботи розподілених файлообмінних сервісів. У даній роботі розглянуто алгоритм прогнозної частини системи динамічного управління трафіком, яка дозволить своєчасно попереджати про подальші значні навантаження мережевого трафіку. Як важливий її компонент, знайдена модель ARIMA з мінімальним числом параметрів, що дозволяє здійснювати достовірні короткострокові прогнози.

Робота [14] присвячена розробці моделей управління доступом до обмежених інформаційно-обчислювальних ресурсів підприємства чи організації. Проведено аналіз недоліків моделей та обгрунтування ресурсного підходу до управління IT-інфраструктурою в умовах недостатності ресурсів. Робота СУІ повинна базуватися на сукупності математичних моделей, які враховують важливість бізнес-процесів, що претендують на використання спільних обмежених IT-ресурсів, схемі вибору і

взаємодії моделей з урахуванням поточної ситуації в ІТ-системі, методах управління доступом до ресурсів в умовах їх обмеженості.

Задачі управління, які вирішуються в СУІ при наданні доступу до обмежених ресурсів, було розподілено за шістьма ієрархічними рівнями, безпосередньо пов’язаними з життєвим циклом управління ІТ-інфраструктурою. Для кожного рівня розглянуто математичні моделі і методи розв’язання. Закладені в основу технології управління ІТ-інфраструктурою методи і моделі управління доступом до обмежених ресурсів, що запропоновані в роботі, а також їх реалізація, дозволяють збільшити ефективність використання обмежених ресурсів ІТ-системи. Адміністратори системи управління ІТ-інфраструктурою можуть отримувати інформацію про стан обладнання, програмного забезпечення елементів підсистем і системи в цілому, попередження про можливі несправності та рекомендації щодо швидкого відновлення працездатності компонентів ІТ-системи. Перевагою є те, що розроблені моделі можуть бути застосовані не тільки при управлінні використанням ІТ-ресурсів, але й на етапах проектування та модернізації ІТ-системи для визначення мінімальної кількості ресурсів, необхідних для підтримки найбільш важливих бізнес-процесів.

В роботі [15] розглянуто алгоритм побудови ARIMA моделі часових рядів на основі методології Бокса-Дженкінса. Приведено опис критеріїв, які використовуються для перевірки стаціонарності часових рядів, вибору оптимальної ARIMA моделі та перевірки її адекватності. Основну увагу акцентовано на реалізації цього алгоритму на мові програмування R. Як приклад, приведено реалізацію короткочасного прогнозування температури повітря на Чорногірському географічному стаціонарі. У цій роботі прогнозування метеоданих базується на ідентифікації параметрів ARIMA моделей часових рядів, які задають інформацію про навколишній стан довкілля. Вони добре описують як стаціонарні, так і нестаціонарні часові ряди. Розглянуто алгоритми побудови цих моделей та їхню програмну реалізацію, для якої використовувалася мова програмування R. Причина її використання полягає в наявності статистичних пакетів для моделювання і

прогнозування часових рядів та в можливості не тільки програмувати власні функції, але в багатьох випадках покращувати вже існуючі.

В роботі [16] запропоновано інформаційну модель системи управління інформаційними технологіями підприємства з точки зору процесу управління проблемами. Запропоновано підхід до організації ефективного управління в рамках методології ITSM (IT Service Management). Проте в публікації не наводиться рішення проблеми правильної класифікації інцидентів, що виникає на етапі організації ефективного управління IT-послугами. Необхідно провести аналіз інформаційних потоків, що надходять від користувачів, провести їх класифікацію з метою вибору типових технологій вирішення проблем, що виникають в роботі систем.

В В роботі [17] розглянута можливість впровадження технології віртуалізації, що дозволяє вирішити багато завдань по вдосконаленню IT-інфраструктури ВНЗ. Рішення, що існують на даний момент для віртуалізації серверних ресурсів, систем зберігання даних, робочих місць клієнта дають можливість істотно поліпшити ефективність використання устаткування і понизити витрати на його обслуговування. Розглянуто можливі варіанти реалізації, проекти і результати.

Разом з наявністю безперечних переваг технологія віртуалізації має ряд недоліків, зв'язаних, наприклад, з неможливістю забезпечити максимальну надійність при консолідації серверів, оскільки між віртуальними машинами немає повної ізоляції. Так, збій в роботі операційної системи ФС приводить до необхідності перезавантажувати всі ВМ і їх застосування. Крім того, ПЗ віртуалізації є достатньо «важким», таким, що вимагає для свого функціонування наявності відповідних серверних конфігурацій. Недоліком контейнерної віртуалізації є те, що всі ВМ вимушені працювати під управлінням однієї і тієї ж версії ОС. Це не дозволяє, наприклад, запустити Linux і Windows на одній машині.

В роботі [18] представлена децентралізована архітектура енергозберігаючої системи управління ресурсами для ЦОД. Визначені проблеми мінімізації споживання енергії при виконання вимог QoS (Quality of service – якість обслуговування) та

сформульовані вимоги для політики розподілу ВМ. Запропоновано три стадії безперервної оптимізації розміщення ВМ: перерозподіл відповідно до поточного використанням декількох системних ресурсів, оптимізації віртуальних мережевих топологій, встановлених між ВМ, та перерозподіл ВМ з урахуванням теплового стану ресурсів. Для першої стадії застосовані евристичні алгоритми, які були оцінені за допомогою розширення для інструмента CloudSim. Один з алгоритмів призвів до зменшення споживання енергії в ЦОД.

Окрім того, політика мінімізації міграцій надає гнучке регулювання SLA встановлюючи відповідні значення порогів утилізації: SLA може бути послаблена, що призводить до збільшення споживання енергії. Політика забезпечує гетерогенність апаратних засобів і ВМ; не потребує інформації про конкретні програми що працюють на ВМ; не залежить від типу навантаження. Отримані результати дослідження показали, що динамічне об'єднання ВМ призводить до значної економії енергії забезпечуючи вимоги якості обслуговування та скорочення витрат на експлуатацію.

Автори в цьому дослідженні [19] розробили та оцінили моделювання прогнозування хмарних клієнтів для базової тестової версії TPCW, використовуючи три методи машинного навчання: Vector Machine Support (SVM), нейронних мереж (NN) та лінійна регресія (LR). Включили показники SLA для часу відповіді та пропускної спроможності до моделі прогнозування з метою надання клієнту більш надійного вибору рішення щодо масштабування. Результати показують, що Support Vector Machine забезпечує найкращу модель прогнозування. В даній статті автори побудували три моделі прогнозування, що використовують лінійну регресію, нейронну мережу та векторну регресію підтримки для дворівневої веб-програми TPCW. Окрім традиційного єдиного метричного прогнозування з використанням ЦП, додали два показники SLA. Час відгуку та пропускну спроможність до моделі прогнозування. Модель підтримки векторної регресії (SVR) показала високу точність прогнозування як з нейронною мережею, так і з лінійною регресією в вікні від 9 до 12 хвилин. З трьох моделей SVR є найбільш переважною моделлю для прогнозування. Отже, хмарні клієнти можуть використовувати SVR для побудови своїх моделей

прогнозування. Крім того, додавання метрик SLA бізнес-рівня (час відгуку та пропускна спроможність) у модель прогнозування відкриває шлях до матриці рішень комбінацій для адаптивного розподілу ресурсів; зокрема збільшення інфраструктури та кількіості віртуальної машини. Автори вважають це дуже корисним, оскільки час відгуку та пропускна спроможність можуть деградувати задовго до того, як програма досягне встановленого порога використання CPU.

В роботі [20], щоб досягти ефективного забезпечення ресурсів у хмарі, автори пропонують метод прогнозування навантаження на кілька кроків, KSwSVR, на основі статистичної технології навчання. Він об'єднує вдосконалений алгоритм SVR і Kalman, і не вимагає доступу до внутрішніх деталей програми. Поліпшений SVR дає більшу вагу для більш важливих даних, ніж стандартний SVR, більш обгрунтовано використовуючи історичну інформацію. Kalman Smoother використовується для усунення шуму даних про використання ресурсів, що походять від помилок вимірювання. Відкриті дані завантаженості ВМ різних ресурсів використовувались для перевірки точності прогнозування, стабільності та пристосованості. Цей статистичний підхід, що ґрунтується на навчанні, не призначений для конкретного об'єкта прогнозування, тому автори вважають, що він буде демонструвати високу продуктивність при роботі з різними об'єктами (програма, ВМ, хост і хмарна система) та ресурсами (обчислення, зберігання та мережа). Експеримент із розподілу процесора показав, що точне прогнозування навантаження KSwSVR може значно зменшити споживання ресурсів при забезпеченні QoS.

У цій роботі автори [21] запропонували нову структуру забезпечення дрібнозернистої еластичності для областей IaaS. Наскільки відомо, ніхто не звернувся до проблеми динамічного забезпечення шляхом пошуку оптимального компромісу між перерозподілом та недостатнім розподілом, використовуючи модель вартості. Автори розглядають пов'язану роботу щодо еластичності та максимізації доходів у хмарах. Велика кількість літератури належить до категорії реактивних планувальників. В роботі [22] розроблено менеджера ресурсів, який вивчає SLA, який має на меті максимізувати дохід постачальників хмарних обчислень. На основі безпосереднього стану системи забезпечується відповідна

кількість ресурсів. Однак, це не зафіксувало б тенденцію робочого навантаження, яку розроблена система зафіксувала за допомогою моделі SARIMA. У [23] запропоновано стохастичну модель ресурсів у віртуалізованих середовищах та алгоритм планування для забезпечення гарантій якості та цілей консолідації серверів. У цій статті та інших роботах, спрямованих на консолідацію серверів та віртуальних машин, еластичність робочого навантаження не враховується. В [24] запровадили алгоритм міграції та консолідації ВМ, в якому автори використовують періодограми, щоб знайти періодичні компоненти в серії використання ЦП. Автори демонструють, що мінливість у робочому навантаженні приносить переваги динамічного розміщення віртуальних машин. Фреймоврк, використовуючи цикл зворотного зв'язку, автоматично вийде з найбільш відповідного типу розподілу на основі навантаження. Якщо на робочому навантаженні немає шаблону, він автоматично забезпечить ресурси, близькі до статичного забезпечення. Якщо навантаженість коливається, тоді система могла б обрати локальні ресурси (вертикальне масштабування) або реплікацію (горизонтальне масштабування) на основі прогнозованого навантаження. Інша пов'язана робота [25] використовує метод прогнозування ансамблевих часових подій, щоб мінімізувати кількість фізичних серверів, проте SLA не виконуються. Автори роботи [26] запропонували схему PRESS (PREDICTIVE Elastic ReSource Scaling) для хмарних систем, де вони знаходять найбільш домінуючу частотну компоненту, якщо в сигналі є значний періодичний шаблон, в іншому випадку вибрати модель Маркова для прогнозування, автори заявили про високу точність прогнозування. Автори вважають, що можуть використовувати будь-яку стратегію прогнозування, оскільки вона передбачається компонентом чорного ящику. Більша точність прогнозування дозволить зменшити загальні витрати на обслуговування ЦОД. Отже, розроблений фреймоворк забезпечує просування до сучасних систем. В роботі [27] описано структуру масштабування еластичних ресурсів, яка спирається на схему PRESS для хмарних систем. Вони використовують стратегії зменшення порушень SLA шляхом додавання додаткової кількості ресурсів на основі різних сценаріїв. У документі

найбільш правильне заповнення ресурсів автоматично виявляється за допомогою оптимального довірчого інтервалу, який мінімізує загальну надлишкову вартість. Крім того, замість прогнозування кожного ресурсу окремо (процесор, пам'ять тощо), прогнозують рівень запиту, який є справжнім показником навантаження і використовується для виведення кортежу ресурсу.

З метою розробки алгоритму для розв'язання проблеми прогнозування ресурсів пропонується застосувати змішану модель прогнозування, а саме поєднати авторегресію, авторегресію ковзного середнього, інтегровану авторегресію ковзного середнього та метод групового урахування аргументів. Цей метод полягає у тому, що на кожному кроці прогнозування, алгоритм обирає модель прогнозування яка дала на попередньому кроці найкращий результат. Метою алгоритма є мінінмізувати похибку прогнозування.

При проведенні аналізу літературних джерел були оцінені основні методи прогнозування потреби ресурсів для серверної системи в умовах хмарного обчислення і виявлені їх недоліки, основними з яких є:

- прогнозування лише за допомогою одного методу або моделі;
- статичне вікно розміру даних для моделей та методів прогнозування.

Метою дослідження є поліпшення точності прогонзування потреби ресурсів для серверної системи в умовах хмарного обчислення шляхом розробки алгоритму прогнозування, що дозволяє збільшити точність прогнозування завантаженості серверної системи.

Для досягнення поставленої мети мають бути виконані наступні завдання:

- проаналізувати предметне прогнозування ресурсів для серверної системи в умовах хмарного обчислення;
- провести огляд методів прогнозування ресурсів для серверної системи;
- розробити алгоритм для прогнозування потреби ресурсів серверної системи;
- виконати програмну реалізацію алгоритму;
- провести дослідження ефективності розробленого алгоритму.

2 ІСНУЮЧІ МЕТОДИ ПРОГНОЗУВАННЯ ПОТРЕБИ РЕСУРСІВ ДЛЯ СЕРВЕРНОЇ СИСТЕМИ

У статистиці під часовим рядом розуміються послідовно виміряні через деякі (найчастіше рівні) проміжки часу дані [28]. Аналіз часових рядів об'єднує методи вивчення часових рядів, які намагаються зрозуміти природу точок даних, так і намагаються побудувати прогноз. Прогнозування часових рядів полягає в побудові моделі для передбачення майбутніх подій грунтуючись на відомих подіях минулого, передбачення майбутніх даних до того, як вони будуть виміряні. Наприклад – прогноз ціни відкриття біржі грунтуючись на попередній її діяльності.

Вибір методу побудови прогнозу по часовому ряду залежить від завдання, як правило, визначаться тим, який прогноз потрібно отримати - довгостроковий або короткочасний. Побудова довготривалого прогнозу є завданням значно складнішим, не завжди має рішення (причому не тільки по технічних причин - в ряді випадків існує так званий «горизонт прогнозу». Горизонт прогнозу – проміжок часу, на який розраховується прогноз.

Економетричний аналіз ґрунтується на вихідних статистичних даних. При цьому, якщо процес реєстрації вихідних статистичних даних відбувається в часі t і саме час фіксується поряд зі значеннями аналізованих характеристик $x_i^j(t_k)$, де змінна j пробігає значення $j = 1,\dots,p$, номер об'єкта і пробігає значення $i = 1,\dots,n$ і $k = 1,2,..,N$, то говорять про статистичний аналізі так званих панельних даних. Якщо зафіксувати номер змінної j і номер статистично обстежуваного об'єкта i, то розташовану в хронологічному порядку послідовність значень називають одновимірним часовим рядом.

$$X_j^i(t_1), X_j^i(t_2), \dots, X_j^i(t_k), \qquad (2.1)$$

Якщо ж одночасно розглядати p одномірних часових рядів виду (2.1), тобто дослідити закономірності у взаємопов'язаномій поведінці часових рядів (2.1) для $j = 1,\dots,p$, характеризують динаміку p змінних, виміряних на якомусь одному

об'єкті, то тоді говорять про статистичномий аналіз багатовимірного часового ряду $X(t) = x_1(t_k), x_2(t_k), \dots, x_p(t_k),\ \ k = 1,2, \dots, N$. Проблема прогнозування полягає в побудові коротко-, середньо- та довгострокових прогнозів. Тим не менше, використання доступних до моменту часу $t = N$ спостережень часового ряду для прогнозування значення $x(t)$ на один або кілька тактів вперед (тобто для прогнозної оцінки значень $x(t_{N+l}), l = 1,2 \dots$) може з'явитися основою для:

- планування в економіці, виробництві, торгівлі;
- управління та оптимізації, що протікають в суспільстві соціально-економічних процесів;
- часткового управління важливими параметрами демографічних процесів та екологічної ніші суспільства;
- прийняття оптимальних рішень в бізнесі.

Принципові відмінності часового ряду від випадкової вибірки полягають у тому, що члени часового ряду не є статистично незалежними і не є однаково розподіленими, тобто $P\{x(t_1) < x\} \neq P\{x(t_2) < x\}$ при $t_1 \neq t_2$

Це означає, що не можна поширювати властивості і правила статистичного аналізу випадкової вибірки на тимчасові ряди. З іншого боку, взаємозалежність членів часового ряду створює свою специфічну базу для побудови прогнозних значень аналізованого показника (тобто для побудови оцінок $\tilde{x}(t_{N+k})$ для невідомих значень $x(t_{N+k})$) по спостережуваним значенням $x(t_1), x(t_2), \dots, x(t_k)$.

Кожен часовий ряд формується під впливом великого числа факторів, які умовно можна поділити на три групи:

- фактори, що формують тенденцію ряду, яка описується за допомогою тієї чи іншої невипадкової функції $f_{\text{тр}}(t)$, як правило, монотонної. Цю функцію називають функцією тренда;
- фактори, що формують сезонні коливання ряду (що повторюються з певною періодичністю (рік, тиждень, добу і т.п.) коливання аналізованої

ознаки), результат дії сезонних чинників позначається за допомогою невипадкової функції $\varphi(t)$;

– фактори, що формують циклічні коливання ряду (зміни аналізованої ознаки, зумовлені дією довготривалих циклів економічної, демографічної або астрофізичної природи). Результат дії циклічних факторів будемо позначати за допомогою невипадковою функції $\psi(t)$;

– випадкові фактори.

Зазвичай в процесі беруть участь не всі фактори одночасно і ніколи сезонні і циклічні коливання разом, так що можна їх об'єднати. Участь випадкових чинників повинна враховуватися завжди, так як вони зумовлюють стохастичну природу елементів ряду.

Кожен член ряду можна представити таким чином:

$$x(t) = \mathrm{X}(A) f_{\text{тр}}(t) + X(B)\varphi(t) + X(C)\psi(t) + \varepsilon(t), \qquad (2.2)$$

де

$$X(D)\begin{cases} 1, & \text{якщо фактори типа } D \text{ формують значення } x(t); \\ 0, & \text{в іншому випадку;} \end{cases}$$

$$D = A, B, C.$$

Висновки про те, беруть участь чи ні фактори даного типу в формуванні значень $x(t)$, можуть базуватися як на аналізі змістовної сутності завдання (тобто бути апріорно-експертними за своєю природою), так і на спеціальному статистичному аналізі досліджуваного часового ряду.

Часові ряди можна розділити на стаціонарні і нестаціонарні. Їх властивості істотно відрізняються, і для моделювання рядів повинні застосовуватися різні методи.

Ряд y_t називається строго стаціонарним, якщо спільний розподіл m спостережень $y_{t_1}, y_{t_2}, y_{t_m}$, не залежить від зсуву за часом, тобто збігається з розподілом y_{t_1+t}, y_{t_2+t}, y_{t_m+t}для будь-яких $m, t, t_1, \ldots, t_m$.

Ряд y_t називається слабо стаціонарним, якщо його середнє, ковариація і дисперсія не залежить від моменту часу t:

$$E(y_t) = \mu < \infty, V(y_t) = \gamma_0, Cov(y_t, y_{t-k}) = \gamma_k, \quad (2.3)$$

Перевірка на стаціонарність. Визначити, [29] чи є ряд стаціонарним, можна кількома способами. Найпростішим способом є побудова графіка отриманих спостережень. У ньому можна виявити тренд або періодичність компоненту (сезонність). Також, можна побудувати графік вибіркової автокорреляційної функцією ACF (autocorrelation function) або коррелограмми (correlogram):

$$\rho_k = \frac{Cov(y_t, y_{t-k})}{V(y_t)} = \frac{\gamma_k}{\gamma\_0}, \quad (2.5)$$

$$r_k = \widehat{\rho_k} = \frac{\sum_{t=k+1}^{n}(y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^{n}(y_t - \bar{y})^2}, k = 1,2, \ldots., , \quad (2.6)$$

де $\bar{y}$ – середнє, $\bar{y} = \sum_{i=1}^{n} y_i$.

Коррелограмм стаціонарного часового ряду «швидко убуває» з ростом k після кількох перших значень. Якщо ж графік убуває досить повільно, то є підстави припустити нестационарность ряду.

Крім цього можна побудувати графік часткової автокореляційної функції (англ. partial autocorrelation function, PACF). PACF(k) є чиста кореляція між y_t та y_{t-k} при виключенні впливу проміжних значень $y_{t-1}, y_{t-2}, \ldots, y_{t-k+1}$. Вона також повинна швидко спадати для стаціонарного процесу. Також можна використовувати формальні тести на наявність одиничного кореня (тест Дікі-Фуллера DF, розширений тест Дікі-Фуллера ADF, тест Маккінлі).

Тренд. Тренд [30] – тенденція зміни показників часового ряду. Тренди можуть бути описані різними функціями - лінійними, статичними, експонентними. Тип тренда встановлюють на основі даних часового ряду, шляхом осереднення показників динаміки ряду, на основі статистичної перевірки гіпотези про сталість

параметрів графіка. Трендом називають конкретний, в формі певної монотонної кривої, опис тенденції розвитку. Розглянемо наступний часовий ряд:

$$y_t = \alpha + \beta t + \varepsilon_t, \qquad (2.7)$$

Він являє собою суму заданої складової $\alpha + \beta t$ (лінійний тренд) і випадкової складової ε_t, яка є стаціонарним часовим рядом з нульовим середнім. Також, можна зустріти квадратичний тренд: $y_t = \alpha + \beta t + \gamma t^2$, експоненціальне: $y_t = \alpha e^{\beta t}$ та інші.

Для того, щоб виділити тренд в моделі (2.3), можна застосувати звичайну техніку оцінювання параметрів регресійних рівнянь, вважаючи t незалежною змінною. Після цього виходить ряд залишків, для опису якого можна застосувати моделі стаціонарних часових рядів.

Сезонність. [30] Часові ряди можуть враховувати сезонну компоненту. Наприклад, в квартальних даних може спостерігатися сезонна компонента з періодом *n*:

$$y_t = S(t) + \varepsilon_t, \quad S(t+n) \equiv S(t), \qquad (2.8)$$

де $S(t)$ – це сезонна компонента;

ε_t – білий шум.

Тут ряд y_t представлений у вигляді композиції періодичної детермінованої складової $S(t)$ і випадкової складової ε_t, яка є стаціонарним часовим рядом з нульовим середнім. Сезонну компоненту $S(t)$ можна представити у вигляді $S(t) = \beta_1 d_{1t} + \beta_2 d_{2t} + \cdots, \beta_n d_{nt}$, де d_i - фіктивні (бінарні) змінні для кварталів. Для виділення сезонної компоненти ми можемо застосувати методи оцінювання параметрів регресій до рівняння:

$$y_t = \beta_1 d_{1t} + \beta_2 d_{2t} + \cdots, \beta_n d_{nt} + \varepsilon_t, \qquad (2.9)$$

Як і в разі виділення тренда, методи моделювання стаціонарних часових рядів застосовуються далі до ряду залишків регресії (2.6)

2.1 Регресійні моделі

Регресійний аналіз [31] - метод моделювання вимірюваних даних і дослідження їх властивостей. Дані складаються з пар значень залежної змінної (змінної відгуку) і незалежної змінної. Регресійна модель є функція незалежної змінної і параметрів з доданою випадковою змінною. Параметри моделі налаштовуються таким чином, щоб модель найкращим чином наближалася до вхідних даних. Критерієм якості наближення (цільовою функцією) зазвичай є середньоквадратична помилка: сума квадратів різниці значень моделі і залежної змінної для всіх значень незалежної змінної як аргумент. Передбачається, що залежна змінна є сума значень деякої моделі і випадкової величини. Щодо характеру розподілу цієї величини робляться припущення, звані гіпотезою породження даних. Для підтвердження або спростування цієї гіпотези виконуються статистичні тести, звані аналізом залишків. При цьому передбачається, що незалежна змінна не містить помилок. Регресійний аналіз використовується для прогнозу, аналізу часових рядів, тестування гіпотез і виявлення прихованих взаємозв'язків в даних.

2.1.1 Авторегресійна модель

Авторегресійна модель (англ. Autoregressive model, AR) [32] – модель часових рядів, в якій значення часового ряду в даний момент лінійно залежать від попередніх значень цього ж ряду. Авторегресійний процес порядку p (AR(p)-процес) визначається наступним чином:

$$X_t = c + \sum_{i=1}^{p} a_i X_{t-i} + \varepsilon_t \,, \qquad (2.10)$$

де X_t – це зпрогнозоване значення на період t;

c – константа, зазвичай для спрощення рівняється нулю;

a_i – параметри моделі, коефіцієнти авторегресії;

ε_t – білий шум.

Найпростішим прикладом є Авторегресійний процес першого порядку AR (1) -процес:

$$X_t = c + a_1 X_{t-1} + \varepsilon_t \,, \qquad (2.11)$$

Для даного процесу коефіцієнт авторегресії збігається з коефіцієнтом автокореляції першого порядку.

Якщо ввести лаговий оператор $L: Lx_t = x_{t-1}$, то авторегресійну модель можна представити таким чином $X_t = c + \sum_{i=1}^{p} a_i L^i X_{t-i} + \varepsilon_t$.

Стаціонарність авторегресійного процесу залежить від коренів характеристичного полінома $a(z) = c + \sum_{i=1}^{p} a_i L^i X_{t-i} + \varepsilon_t$

2.1.2 Авторегресійна модель ковзного середнього

У статистичному аналізі часових рядів моделі авторегресії — ковзного середнього [33] (англ. Autoregressive moving-average model, ARMA) пропонують економний опис (слабко) стаціонарного стохастичного процесу в термінах двох многочленів, одного для авторегресії, а другого — для ковзного середнього. Загальну модель ARMA було описано в 1951 році в дисертації Пітера Уіттла «Перевірка гіпотез в аналізі часових рядів» і популяризовано в книзі Джорджа Бокса та Ґвилима Дженкінса 1970 року.

Для заданого часового ряду даних X_t модель ARMA є інструментом для розуміння та передбачування майбутніх значень цього ряду. Ця модель складається з двох частин: авторегресійної, та ковзного середнього. Частина AR передбачає регресування цієї змінної за її власними запізнюваними (тобто, минулими) значеннями. Частина MA передбачає моделювання члену похибки як лінійної комбінації членів похибки, що стаються в поточний момент та в різні моменти часу в минулому. На цю модель зазвичай посилаються як на модель $ARMA(p, q)$, де p —

порядок авторегресійної частини, а q — порядок частини ковзного середнього (як визначено нижче).

Позначення $AR(p)$ стосується авторегресійної моделі порядку p. Модель $AR(p)$ записують як (2.1)

Позначення $MA(q)$ стосується моделі ковзного середнього порядку q:

$$X_t = \mu + \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \,, \qquad (2.12)$$

де X_t – це зпрогнозоване значення на період t;

μ – математичне сподівання;

α – дійсні числа, авторегресійні коефіцієнти;

θ_i – дійсні числа, коефіцієнти змінного середнього;

ε_t – білий шум.

Позначення $ARMA(p, q)$ стосується моделі з p авторегресійними членами та q членами ковзного середнього. Ця модель містить моделі $AR(p)$ та $MA(q)$:

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} a_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \,, \qquad (2.13)$$

Члени похибки ε_t, як правило, вважають незалежними однаково розподіленими випадковими величинами (РВВ), відбираними з нормального розподілу з нульовим середнім: $\varepsilon_t \sim N(0, \sigma^2)$, де σ^2 – дисперсією. Ці припущення може бути послаблено, але це змінить властивості моделі. Зокрема, зміна припущення про РВВ призведе до принципової відмінності.

Пристосовування моделей. Пошук відповідних значень p та q в моделі ARMA(p, q) може бути полегшено шляхом побудови частинних автокореляційних функцій задля оцінки p, а також використання автокореляційних функцій задля оцінки q. Додаткову інформацію можливо підбирати, розглядаючи ті ж функції для залишків моделі, пристосованої початковим вибором p та q. Моделі ARMA після вибору p та q загалом не може бути пристосовувано за допомогою регресії найменших квадратів задля знаходження значень параметрів, які мінімізують член

похибки. Загалом доброю практикою вважають знаходити найменші значення p та q, які забезпечують прийнятну пристосованість до даних.

2.1.3 Авторегресійне інтегроване ковзного середнього

Підхід авторегресійного інтегрованого ковзного середнього (англ. autoregressive integrated moving average, ARIMA) [34] до часових рядах полягає в тому, що в першу чергу оцінюється стационарность ряду. Різними тестами виявляються наявність поодиноких коренів і порядок інтегрованості часового ряду (зазвичай обмежуються першим або другим порядком). Далі, при необхідності, (якщо порядок інтегрованості більше нуля) ряд перетворюється взяттям різниці відповідного порядку і вже для перетвореної моделі будується деяка ARMA-модель, оскільки передбачається, що отриманий процес є стаціонарним, на відміну від вихідного нестаціонарного процесу (разностно-стаціонарного або інтегрованого процесу порядку d).

ARIMA - модель і методологія аналізу часових рядів. Є розширенням моделей ARMA для нестаціонарних часових рядів, які можна зробити стаціонарними взяттям різниць деякого порядку від вихідного часового ряду (так звані інтегровані або разностно-стаціонарні часові ряди). Модель $ARIMA(p, d, q)$ означає, що різниці часового ряду порядку d належать моделі $ARMA(p, q)$.

Модель $ARIMA(p, d, q)$ для нестаціонарного часового ряда X_t має вигляд:

$$\Delta^d X_t = c + \varepsilon_t + \sum_{i=1}^{p} a_i \Delta^d X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \,, \qquad (2.15)$$

де X_t – це зпрогнозоване значення на період t;

c – константа, зазвичай для спрощення рівняється нулю;

a_i – параметри моделі, коефіцієнти авторегресії;

a_i – параметри моделі, ковзного середнього;

$\Delta\hat{}d$ – оператор різниці часового ряду порядку d (послідовне взяття d раз різниць першого порядку - спочатку від тимчасового ряду, потім від отриманих різниць першого порядку, потім від другого порядку і т.д.)

ε_t – білий шум.

Також, дана модель інтерпретується як $ARMA(p+d,q)$ модель з d одиничними розв'язками. При $d = 0$ маємо звичайну ARMA модель.

ARIMA-моделі дозволяють моделювати інтегровані або разностно-стаціонарні часові ряди (DS-ряди, diference stationary). Часовий ряд називається інтегрованим порядку k, якщо різниці ряду порядку $\Delta^k x_t$, тобто є стаціонарними, в той час як різниці меншого порядку (включаючи нульового порядку, тобто сам тимчасової ряд) не є стаціонарними щодо деякого тренда рядами (TS-рядами, trend stationary). В зокрема $I(0)$ – це стаціонарний процес. Порядок інтегрованості часового ряду і є порядок d моделі.

Методологія Бокса-Дженкінса. Методологія Бокса-Дженкінса підбору ARIMA моделі для даного ряду спостережень складається з 5 кроків.

Крок 1. Отримання стаціонарного ряду. Ми тестуємо ряд на стаціонарність, використовуючи описані вище методи: візуальний аналіз графіка, візуальний аналіз ACF і PACF, тести на одиничні коріння. Якщо виходить стаціонарний ряд, то переходимо до наступного пункту, якщо немає стаціонарності ряду, то застосовуємо оператор взяття послідовної різниці і повторюємо тестування. Напрактиці послідовна різниця береться, як правило, не більше двох разів.

Крок 2. Після того, як отримано стаціонарний часовий ряд, будуються його вибіркові ACF і PACF, які, як було показано вище, є своєрідними «відбитками пальців» ARMA(p, q) процесу і дозволяють сформулювати кілька гіпотез про можливі порядки авторегресії p і змінного середнього q. Зазвичай рекомендується використовувати моделі можливо більш низького порядку, як правило, з $p\ q$ 3 (якщо немає сезонної компоненти).

Крок 3. Для кожної з обраних на першому етапі моделей оцінюються їх параметри і обчислюються залишки.

Крок 4. Кожна з моделей перевіряється, наскільки вона відповідає даним. З моделей, адекватних даними, вибирається найпростіша модель, тобто з найменшою кількістю параметрів.

Крок 5. Прогнозування. Після того, як обрана модель, можна будувати прогноз на один або кілька кроків за часом і оцінювати довірчі інтервали прогнозованих значень.

2.2 Нейронні мережі

Сучасні цифрові обчислювальні машини здатні з високою швидкодією і точністю вирішувати формалізовані завдання з цілком певними даними за заздалегідь відомими алгоритмами. Проте в тих випадках, коли завдання не піддається формалізації, а вхідні дані неповні, зашумлені або суперечливі, застосовують технології нейронних мереж. Сильною стороною цих комплексів є нестандартний характер обробки інформації. Вона кодується і запам’ятовується не в окремих елементах пам’яті, а в розподілі зв’язків між нейронами. Стан кожного окремого нейрона визначається станом багатьох інших нейронів, пов’язаних з ним. Отже, втрата одного або декількох зв’язків не робить істотного впливу на результат роботи системи в цілому, що забезпечує її високу надійність [35].

Висока «природна» перешкодостійкість і функціональна надійність стосуються як спотворених (зашумлених) потоків інформації, так і в сенсі відмов окремих процесорних елементів. Цим забезпечуються висока оперативність і достовірність обробки інформації, а просте донавчання і перенавчання мереж дозволяють при зміні зовнішніх чинників своєчасно здійснювати перехід на новий рівень вирішуваних завдань.

Штучна нейронна мережа (ШНМ) - математична модель, а також її програмне або апаратне втілення, побудована за принципом організації та функціонування біологічних нейронних мереж - мереж нервових клітин живого організму. Це поняття виникло при вивченні процесів, що протікають в мозку, і при спробі змоделювати ці процеси. Першою такою спробою були нейронні мережі У.

Маккалок і У. Питтса [36]. Після розробки алгоритмів навчання одержувані моделі стали використовувати в практичних цілях: в задачах прогнозування, для розпізнавання образів, в задачах управління та ін.

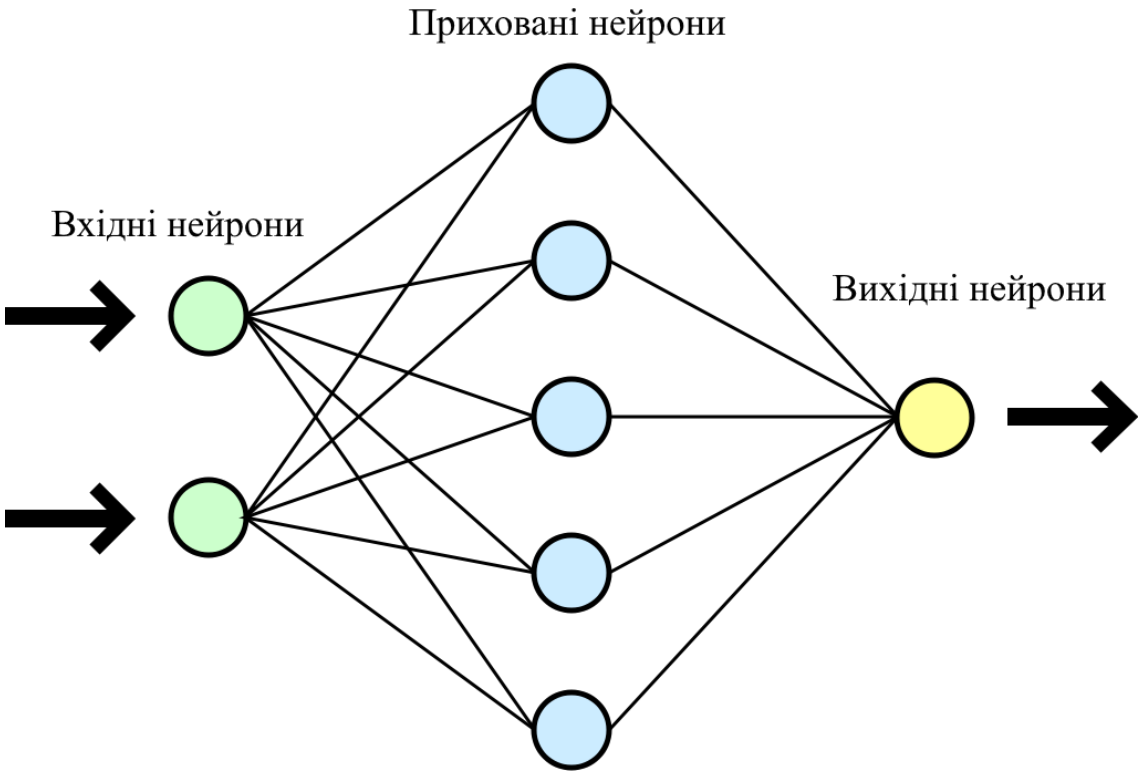


Рисунок 2.1 – Загальний вид мережі

Нейронні мережі не програмуються в звичному сенсі цього слова, вони навчаються. Можливість навчання - одне з головних переваг нейронних мереж перед традиційними алгоритмами. Технічно навчання полягає в знаходженні коефіцієнтів зв'язків між нейронами. В процесі навчання нейронна мережа здатна виявляти складні залежності між вхідними даними і вихідними, а також виконувати узагальнення. Це означає, що в разі успішного навчання мережа зможе повернути вірний результат на підставі даних, які були відсутні в навчальній вибірці, а також неповних і / або «зашумлених», частково спотворених даних.

Нейронні мережі були розроблені на основі функціонування нервової клітини. Основою методу було те, що нейрон досить легко уявити в машинному вигляді, а всю складність нервової клітини і інші якості її роботи можна визначити зв'язками.

Це обумовлює застосування нейронних мереж для різних видів завдань навчання по прецедентах, в тому числі розпізнавання, класифікації, регресії і прогнозування.

Нейронна мережа складається з найпростіших елементів - нейронів. Кожен нейрон має групу синапсів - односпрямованих вхідних зв'язків, з'єднаних з виходами других нейронів, а також має аксон - вихідну зв'язок даного нейрона, з якої сигнал надходить на синапси наступних нейронів. Кожен синапс характеризується величиною синаптичного зв'язку або її вагою w_i .Загальний вид нейрона наведено на рисунку 2.2.

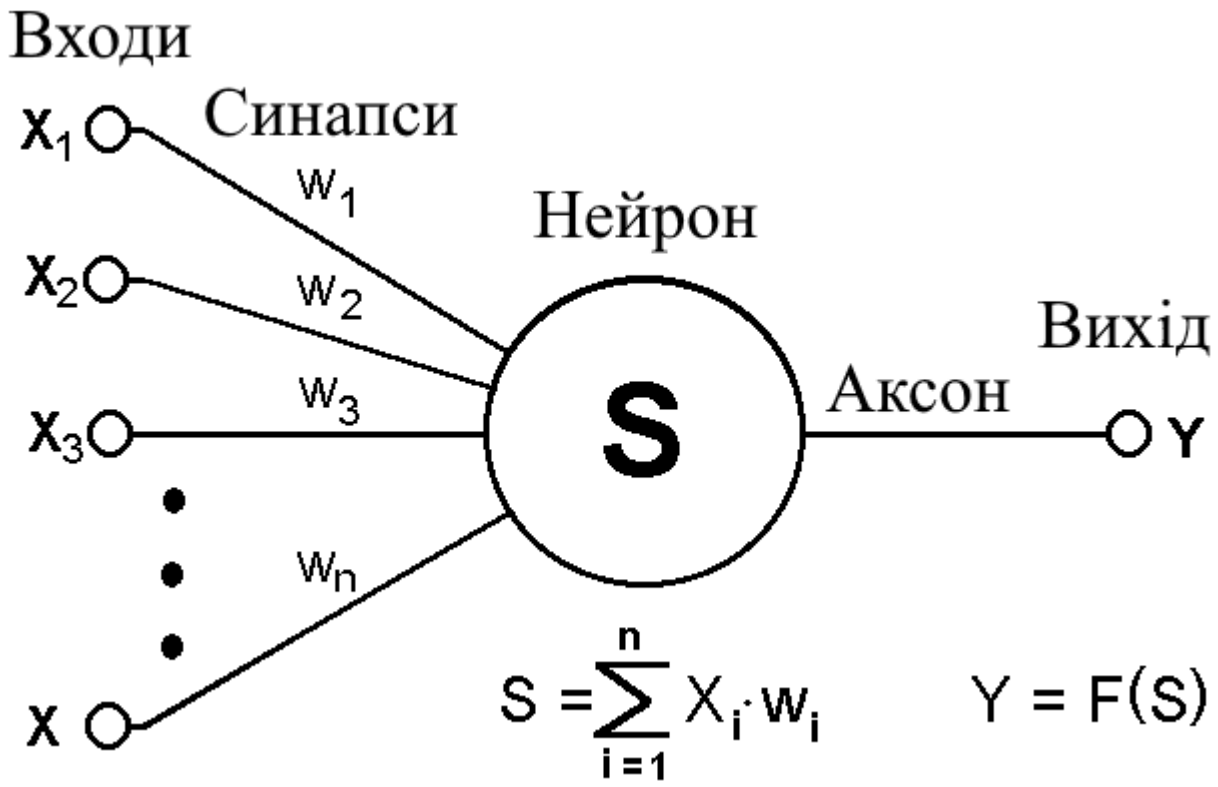


Рисунок 2.2 – Загальний вид нейрона

Нейрон характеризується поточним станом і володіє групою синапсів - односпрямованих вхідних зв'язків, з'єднаних з виходами інших нейронів.

Аксон - вихідний зв'язок даного нейрона, з якої сигнал (збудження або гальмування) надходить на синапси наступних нейронів.

Кожен синапс характеризується величиною синаптичного зв'язку (її вагою w_i).

Поточний стан нейрона визначається як зважена сума його входів:

$$S = \sum_{i=1}^{n} x_i \, w_i$$

Завдання синтезу багатовимірної (MIMO) моделі досить ефективно може бути вирішене за допомогою таких традиційних нейронних мереж, як багатошаровий персептрон (MLP) або радіально-базисна нейронна мережа (RBFN) [37]. Тут, однак, виникає цілий ряд істотних проблем, пов'язаних з характером оброблюваних даних. Справа в тому, що часові ряди, як правило, істотно нестаціонарні, містять нелінійні нерегулярні тренди, різкі скачки і викиди.

Тому, даних, що характеризують стаціонарні ділянки, просто недостатньо, щоб навчити досить великий набір синаптичних ваг багатошарового персептрона або радіально-базисної мережі.

Скоротити кількість параметрів синаптичних ваг можна, використовуючи метод групового урахування аргументів (МГУА), запропонований А.Г. Івахненко [38, 39] і покладений в основу так званої МГУА-нейронної мережі [40].

На рисунку. 2.3 наведено приклад МГУА-нейронної мережі з чотирма входами, одним виходом і нейронами N-A, які є по суті нелінійними адалін (N-Adaline).

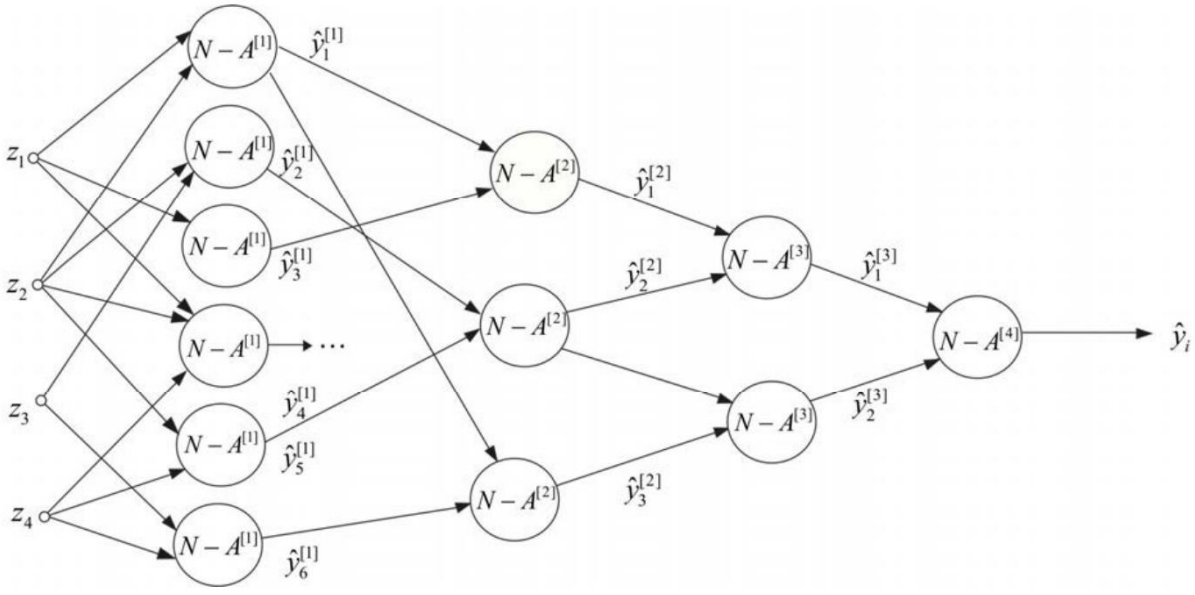


Рисунок 2.3 – МГУА нейронна мережа

2.2.1 Метод групового урахування аргументів

Метод групового урахування аргументів (англ. Group Method of Data Handling, GMDH) - метод породження і вибору регресійних моделей оптимальної складності. Під складністю моделі в GMDH [41] розуміється число параметрів. Для породження використовується базова модель, підмножина елементів якої має входити в шукану модель. Для вибору моделей використовуються зовнішні критерії, спеціальні функціонали якості моделей, обчислені на тестовій вибірці.

GMDH рекомендується до використання в тому випадку, коли вибірка містить кілька елементів. Тоді при побудові регресійних моделей використовувати статистичні гіпотези про щільність розподілу, наприклад, гіпотезу про гауссовский розподілі, неможливо. Тому використовується індуктивний підхід, згідно з яким послідовно породжуються моделі зростаючої складності до тих пір, поки не буде знайдений мінімум деякого критерію якості моделі. Цей критерій якості називається зовнішній критерій, так як під час налаштування моделей і при оцінці якості моделей використовуються різні дані. Досягнення глобального мінімуму зовнішнього критерію при породженні моделей означає, що модель, що доставляє такий мінімум, є шуканою.

Один з авторів цього методу А.Г. Івахненко пише: «Здійснюється цілеспрямований перебір багатьох моделей-претендентів різної складності по ряду критеріїв. В результаті знаходиться модель оптимальної структури у вигляді одного рівняння або системи рівнянь. Мінімум критерію селекції визначає модель оптимальної структури».

Опис алгоритму GMDH. [42] Індуктивний алгоритм відшукання моделі оптимальної структури складається з наступних основних кроків.

Нехай задана виборка $D = \{(x_n, y_n)\}_{n=1}^{N}, x \epsilon \mathcal{R}^m$. Виборка розбивається на навчальну та тестову. Позначаємо l, C – множина індексів з $\{1, \dots, N\} = W$. Ці множини задовольняють умовам розбивання $l \cup C = W, l \cap C = \emptyset$. Матриця X_l складається з тих векторів-рядків x_n, для яких індекс $n \in l$. Вектор Y_l складається Y_n, для яких індекс $n \in l$. Розбивання виборки надається у вигляді:

$$X_w = \left(\frac{X_l}{X_c}\right), y_w = \left(\frac{y_l}{y_c}\right), y_w \in \mathcal{R}^{N\times 1}, X_w \in \mathcal{R}, |l| + |C| = N\ , \qquad (2.16)$$

Назначається базова модель. Ця модель описує відношення між залежною змінною y і вільною змінною x. Наприклад, використовується функціональний ряд Вольтера, називаємий також поліномом Колмогорова-Габора:

$$y = w_0 + \sum_{i=1}^{m} w_i x_i + \sum_{i=1}^{m}\sum_{j=1}^{m} w_{ij} x_i x_j + \cdots\ , \qquad (2.17)$$

де $x = \{x_i | i = 1, \dots, m\}$ – множина вільних змінних;

w – вектор параметрів, вагових коефіцієнтів,

$$w = \langle w_i, w_{ij}, \dots, |i, j, k \dots = 1, \dots, m\rangle. \qquad (2.18)$$

В деяких випадках має сенс збільшити число параметрів вектора вільної змінної x за рахунок додавання нелінійних перетворень окремих змінних. Наприклад, задано кінцева безліч нелінійних функцій $G = \{g|\mathcal{R} \to \mathcal{R}\}$. Додаткова вільна змінна виходить шляхом застосування деякого перетворення з G до однієї або до декількох змінних з безлічі $\{x\}$. Базова модель лінійна щодо параметрів w та нелінійна щодо вільних змінних x.

Виходячи з поставлених завдань вибирається цільова функція - зовнішній критерій, що описує якість моделі. Нижче описані кілька часто використовуваних зовнішніх критеріїв.

Індуктивно породжуються моделі-претенденти. При цьому вводиться обмеження на довжину полінома базової моделі. Наприклад, ступінь полінома базової моделі не повинно перевищувати вказану кількість $\mathcal{R}$. Тоді базова модель подана в вигляді лінійної комбінації заданого числа F_0 добутків вільних змінних:

$$y = f(x_1, x_2, \dots, x_1^2, x_1 x_2, x_2^2, \dots, x_m^{\mathcal{R}}), \qquad (2.19)$$

тут f - лінійна комбінація. Аргументи цієї функції перевизначаються наступним чином: $x_1 \to a_1, x_2 \to a_2, \dots, x_1^2 \to a_\alpha, x_1 x_2 \to a_\beta, x_2^2 \to a_\gamma, \dots, x_m^q \to a_{F_0}$, тобто,

$$y = f\left(a_1, a_2, \dots, a_{F_0}\right). \qquad (2.20)$$

Для лінійно вхідних коефіцієнтів задається одноіндексна нумерація $w = \langle w_1, \dots, w_{F_0} \rangle$. Тоді модель може бути представлена у вигляді лінійної комбінації

$$y = w_0 + \sum_{i=1}^{F_0} w_i a_i = w_0 + w \cdot a. \quad (2.21)$$

Кожна породжувана модель задається лінійною комбінацією елементів $\{(w_{i,} a_i)\}$, в якій безліч індексів $\{i\} = s$ є підмножиною $\{1, \dots, F_0\}$.

Налаштовуються параметри моделей. Для настройки використовується внутрішній критерій - критерій, який вираховується з використанням навчальної вибірки. Кожному елементу вектора x_n – елемента вибірки D ставиться у відповідність вектор a_n, алгоритм побудови відповідності зазначений вище. Будується матриця A_W - набір векторів-стовпців a_i. Матриця A_W розбивається на підматріци A_l і A_C. Найменшу невязку $|y - \bar{y}|$, де $\bar{y} = A\bar{w}$ доставляє значення вектора параметрів $\bar{w}$, який обчислюється методом найменших квадратів:

$$\bar{w}_G = (A_G^T A_G)^{-1} A_G^T y_G \quad (2.22)$$

де $G \in \{l, C, W\}$.

При цьому в якості внутрішнього критерію виступає середньоквадратична помилка

$$\varepsilon_G^2 = |y_G - A_G \bar{w}_G|^2 \quad (2.23)$$

Відповідно до критерію $\varepsilon_G^2 \to min$ відбувається настройка параметрів w і обчислення помилки на тестовій підвибірки, позначеної G, тут $G = l$. При ускладненні моделі внутрішній критерій не дає мінімуму для моделей оптимальної складності, тому для вибору моделі він не придатний.

Для вибору моделей обчислюється якість породжених моделей. При цьому використовується контрольна вибірка і призначений зовнішній критерій. Помилка на підвибірки H позначається

$$\Delta^{2(H)} = \Delta^2(H \backslash \mathrm{G}) = |y_H - A_H \overline{w}_G|^2, \qquad (2.24)$$

де $H \in \{l, C\}, H \cap G = \emptyset$.

Це означає, що помилка обчислюється на підвибірки H при параметрах моделі, отриманих на підвибірки G.

Модель, що доставляє мінімум зовнішнім критерієм, вважається оптимальною. Якщо значення зовнішнього критерію не досягає свого мінімуму при збільшенні складності моделі або значення функції якості незадовільно, то вибирається найкраща модель з моделей заданої складності. Під складністю моделі мається на увазі число параметрів, що настроюються моделі. Існують наступні причини, за якими глобальний мінімум може не існувати:

- дані занадто зашумлені;
- серед даних немає необхідних для відшукання моделі змінних;
- невірно заданий критерій вибору;
- при аналізі часових рядів існує значна часова затримка на пошук причинно-наслідкового зв'язку.

Зовнішні критерії GMDH. [43] Авторами методу розглянуто досить велике число різних критеріїв вибору моделей. Значна частина цих критеріїв опублікована на сайті [42].

Критерій вибору моделі може бути названий зовнішнім, якщо він отриманий за допомогою додаткової інформації, що не міститься в даних, які використовувалися при обчисленні параметрів моделей. Наприклад, така інформація міститься в додатковій, тестовій вибірці.

Алгоритм GMDH використовує і внутрішній критерій і зовнішній. Внутрішній критерій використовується для налаштування параметрів моделі, зовнішній критерій використовується для вибору моделі оптимальної структури. Можливий вибір моделей за кількома зовнішніми критеріями.

Критерій регулярності. [44] Критерій регулярності $\Delta^2(C)$ включає середнє квадратичне відхилення на навчальної підвибірки C отриману при параметрах моделі, налаштованих на тестовій підвибірки l.

$$\Delta^2(C) = |y_C - A_C \widehat{w}_l|^2,$$

де $\widehat{w}_l = (A_l^T A_l)^{-1}(A_l^T y_l)$

Інші модифікації критерію регулярності:

$$\Delta^2(C) = \frac{|y_C - A_C \widehat{w}_l|^2}{|y_C|^2},$$

Критерий $\Delta^2(C)$ також позначається як $\Delta^2(C \backslash l)$, тобто помилка на підвибірки C, при параметрах, отриманих на підвибірки l.

Критерій absolute noise-immune. Стверджується, що за допомогою цього критерію, з сильно зашумлених даних можливо знайти приховані фізичні закономірності.

$$V^2 = (A_W \widehat{w}_l - A_W \widehat{w}_W)^T (A_W \widehat{w}_W - A_W \widehat{w}_C)$$

де $\widehat{w}_W$ – вектор коефіцієнтів, отриманий з виборки W.

Алгоритм породження моделей GMDH. [45] Ціллю GMDH є отримання моделі в результаті перебору моделей з індуктивно-породжуваної безлічі. Параметри кожної моделі налаштовуються так, щоб доставити мінімум обраного зовнішнього критерію. Розрізняють два основних типи алгоритмів GMDH – однорядний і багаторядний.

Всі алгоритми GMDH відтворюють схему масової селекції: послідовно породжуються моделі зростаючої складності. Кожна модель настроюється, методом найменших квадратів знаходяться значення параметрів. З моделей-претендентів вибираються кращі відповідно до обраного критерію. Багаторядні алгоритми можуть обчислювати залишки регресійних моделей після кожного ряду селекції або не вирахував; при цьому використовуються вихідні дані.

Кожна поліноміальна модель однозначно визначається набором індексів s, що входять до неї одночленів.

$$y = a_0 + w(s)a(s),$$

w – вектор коефіцієнтів при одночлені полінома Колмогорова-Габора;

a – вектор результат множення вільних змінних та відповідних одночленів.

$S \subseteq \{1, \dots, F_0\}$ є індекси одночленів, що входять в модель.

При обмеженні ступеня полінома числом R, число одночленів полінома дорівнює

$$F_0 = \sum_{r=1}^{R} \bar{C}_r^P = \sum_{r=1}^{R} \frac{(r+P-1)!}{P!(r-1)!},$$

де $\bar{C}_r^P$ – число комбінацій з повторенням з P по r;

P – число вільних змінних – елементів вектора x

Комбінаторний алгоритм. Комбінаторний (однорядний) [46] алгоритм використовує тільки один ряд вибору. При цьому породжуються всі можливі лінійні комбінації обмеженої складності. Так як під складністю розуміється число лінійно вхідних параметрів w, то складність не перевищує задане значення F_0. Нехай

$$y = w_0 + w_1 a_1 + w_2 a_2 + \cdots + w_{F_0} a_{F_0},$$

Алгоритм виконує наступні кроки. Для всіх комбінацій вхідних аргументів будуються моделі-претенденти неспадкової складності. Параметри кожної моделі налаштовуються методом найменших квадратів за навчальною вибіркою. Найкраща модель вибирається виходячи з мінімуму значення зовнішнього критерію. Як варіант - призначається поріг і вибираються кілька моделей, значення критерію для яких не перевищує цей поріг. При програмуванні даного алгоритму зручно ввести змінну вибору – вектор $c(s) = \langle c_1, \dots, c_{F_0} \rangle$. Елемент $c_i \in \{0, 1\}$ приймає значення 1, якщо $i \in S$, в іншому випадку 0. Тоді модель має вигляд:

$$y = c(s)wa,$$

Так як при породженні моделей необхідно вибирати з 2^{F_0} моделей, що може спричинити неприпустимо великий час обчислень, запропоновані кілька евристичних алгоритмів, що дозволяють скоротити час обчислень, без зменшення максимальної складності моделей.

Багаторядний алгоритм. На першому ряді алгоритму [47] породження моделей задано безліч з F_0 змінних a_i. Породжуються моделі як лінійні комбінації всіляких пар змінних

$$y_{ij} = w_0 + w_i a_i + w_j a_j, i, j = 1, \dots, F_0, i \neq j,$$

Число моделей першого ряду M_1 є число поєднань $C_2^{\{F_0\}} = \frac{1}{2} F_0 (F_0 - 1) = M_1$. Кожна модель, породжувана на ряді, задається парою індексів (i, j). Після породження моделей їх параметри налаштовуються з використанням внутрішнього критерію. Потім вибираються F_1 найкращих моделей з використанням зовнішнього критерію. Ці моделі використовуються в наступному ряді. Безліч обраних моделей задано безліччю пар індексів $\{(i, j)\}$. На першому ряді індекси обраних моделей i, j належать безлічі $s_1 \subset \{1, \dots, F_0\}$.

На кожному наступному ряді нові моделі - породжуються як суми всіляких пар обраних моделей попереднього ряду. Породжені моделі знову налаштовуються, вибираються найкращі. Зупинка породження моделей на наступних рядах відбувається в тому випадку, коли зі збільшенням номера шару, тобто, з ускладненням моделей, відбувається збільшення зовнішнього критерію кращої моделі.

2.3 Висновок до розділу

У даному розділі були досліджені існуючі методи прогнозування. До таких методів було віднесено:

- авторегресійна модель;
- авторегресійна модель ковзного середнього;
- авторегресійне інтегроване ковзного середнього;
- метод групового урахування аргументів.

3 РОЗРОБКА АЛГОРИТМУ ГІБРІДНОГО ПРОГНОЗУВАННЯ ПОТЕРБИ РЕСУРСІВ ДЛЯ СЕРВЕРНОЇ СИСТЕМИ

3.1 Гібрідний алгортим прогнозування потреби ресурсів для серверної системи на основі нейронної мережі та авторегресійних моделей

Хмарні обчислення – модель зручного мережевого доступу на вимогу до деякого загального фонду обчислювальних ресурсів (наприклад, мереж передачі даних, серверів, пристроїв зберігання даних, додтаків і сервісів), які можуть бути оперативно надані та звільнені з мінімальними експлуатаційними витратами або зверненнями до провайдера.

Споживачі хмарних обчислень можуть значно зменшити витрати на інфраструктуру інформаційних технологій (в короткостроковому і середньостроковому планах) і гнучко реагувати на зміни обчислювальних потреб, використовуючи властивості обчислювальної еластичності хмарних послуг.

Під час роботи ВМ, на них постійно збільшується та зменшується навантаження, відповідно, це навантаження змінюється і на серверній системі. Щоб запобігти перевантаження або недовантаження серверної системи потрібно постійно мати актуальний та точний прогноз утилізації ресурсів в наступний період часу. Адже коли система недовантажена, йде простій ресурсів, які можна було б використати для інших ВМ. А коли перевантажена – відбуваєтсья затримка доступу до сервісу, тобто порушення SLA. SLA виконується для кожного клієнта, коли вся продуктивність, яку потребуть застосунки всередині ВМ, забезпечується в будь який час.

Алгоритм прогнозування на основі моделей аналізу часвоих рядів наведено в додатку А, на плакаті блок схема гібрідного алгоритму прогнозування.

Завдання прогнозування полягає в тому, щоб за наявним спостереженнями часового ряду передбачити майбутні значення утилізації ресурсів. Прогнозування потреби ресурсів для серверної системи грає дуже велику роль, оскільки воно є раціональною основою ждя прийняття рішень керування. Методів прогнозування існує безліч, але одному й тому ж часовому ряді вони обчислють прогноз з різною

похибкою MAPE (3.1) та RMSE (3.2). Тому, під час прийняття рішення управління, треба обирати такий метод, який дає найменшу похибку прогнозу. В дослідженні використовують похибки MAPE та RMSE, які визначаються наступним чином:

$$MAPE_j = \frac{1}{n}\sum_{i=1}^{n}\frac{|x-\hat{x}_{i,j}|}{x}, \quad (3.1)$$

$$RMSE_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x-\hat{x}_{i,j}\right)^2} \quad (3.2)$$

де n – кількість замірів;

x_i – фактичне навантаження;

$\hat{x}_{i,j}$ – прогнозоване навантаження j – го методу;

j є {AR, ARMA, ARIMA, GMDH}.

Алгоритм гібрідного прогнозування має наступні етапи:

Крок 1. Отримання нових даних;

Крок 2. Перевірка чи чи заповнене вікно;

Крок 3. Обчислення прогнозу методом AR;

Крок 4. Обчислення прогнозу методом ARMA;

Крок 5. Обчислення прогнозу методом ARIMA;

Крок 6. Обчислення прогнозу методом GMHD;

Крок 7. Обираємо метод, який дав найменшу похибку на попередньому кроці прогнозування;

Крок 8. Обчислення похибок кожним методом;

Крок 9. Збережння результатів;

Крок 10. Зсув ковзного вікна;

Під час роботи алгоритму дані у вікні прогнозування (часовий ряд, який є вхідними даними для обчислення прогнозу) оновлюються з приходом наступних вимірів моніторингу за методом FIFO. Таким чином, розмір ковзного вікна прогнозування залишається постійним. Якщо фактичні дані зявились, алгоритм починає обчилюсвати похибку прогнозів з урахуванням останніх даних.

3.2 Висновок до розділу

В даному розділі наведено постановку досліджуваної задачі обрахування прогнозування потреби ресурсів для серверної системи.

Розроблено метод гібрідного прогнозування потерби ресурсів з використанням методів AR, ARMA, ARIMA, GMDH.

4 ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Інформаційне забезпечення

4.1.1 Вхідні дані

Вхідними даними є:

– інтервал часу трейсу завантаженості ВМ;

– показники завантаженості ВМ, а саме показники використання процесора.

Показники продуктивності містяться в n файлах (n – кількість ВМ) з розширенням .csv в наступному форматі:

$$\begin{matrix} t_1 & cpu_1 \\ t_2 & cpu_2 \\ t_i & cpu_i \\ \dots & \dots \\ t_n & cpu_n \end{matrix},$$

де n – кількість замірів;

$cpu_i \in [0, 100]$ – показник завантаженості CPU в % за інтервал часу $(t_{i-1}, t_i]$;

t_i – момент часу, в який знято показник завантаженості CPU;

Кожен файл містить n рядків продуктивності ВМ, завантаженості CPU та періодом, коли ця завантаженість була. Стовпці рядка розділені за допомогою $";\backslash t"$.

На рисунку 4.1 наведено фрагмент вмісту CSV файлу з вхідними даними.

	A	B
1	Timestamp [ms]	CPU usage [%]
2	1376314846	93.23333333333333
3	1376315146	93.05
4	1376315446	89.15
5	1376315746	90.05
6	1376316046	93.56666666666666
7	1376316346	93.25
8	1376316646	92.75
9	1376316946	86.78333333333335
10	1376317246	89.51666666666667
11	1376317546	95.08333333333334
12	1376317846	94.25
13	1376318146	55.41666666666667
14	1376318446	0.7333333333333333
15	1376318746	0.6
16	1376319046	0.65
17	1376319346	0.5833333333333334
18	1376319646	0.5333333333333333
19	1376319946	0.6166666666666667
20	1376320246	0.6666666666666667
21	1376320546	0.5833333333333334
22	1376320846	0.5833333333333334
23	1376321146	0.5833333333333334
24	1376321446	0.65
25	1376321746	0.5833333333333334
26	1376322046	0.55
27	1376322346	0.55
28	1376322647	0.5
29	1376322947	0.5666666666666667
30	1376323247	0.55
31	1376323547	0.5
32	1376323847	0.5333333333333333

Рисунок 4.1 – Фрагмент вмісту CSV файлу з вхідними даними

4.1.2 Вихідні дані

Вихідними даними є:

– поточний час прогнозування;

– спрогнозоване значення завантаженості процесора кожним алгоритмом;

– похибка MAPE

– похибка RMSE;

– фактичне значення завантаженості CPU.

Вихідні дані містяться у файлі з розширенням .csv в наступному форматі:

$$\begin{array}{ccccc} t_1 & \hat{x}_{1,j} & x_1 & RMSE_{1,j} & MAPE_{1,j} \\ t_2 & \hat{x}_{2,j} & x_2 & RMSE_{2,j} & MAPE_{2,j} \\ \dots & \dots & \dots & \dots & \dots \\ t_n & \hat{x}_{n,j} & x_n & RMSE_{n,j} & MAPE_{n,j} \end{array}$$

де $j \in \{AR, ARMA, ARIMA, GMDH\}$ – алгоритм прогнозування;

$i = \overline{1,n}$;

n – **кількість періодів, кроків прогнозування, один період дорівнює 5хв;**

t_i – інтервал часу, в який знято показник завантаженості CPU;

$\hat{x}_{i,j}$ – спрогнозоване значення завантаженості CPU i-го інтервала часу j-тим алгоритмом прогнозування;

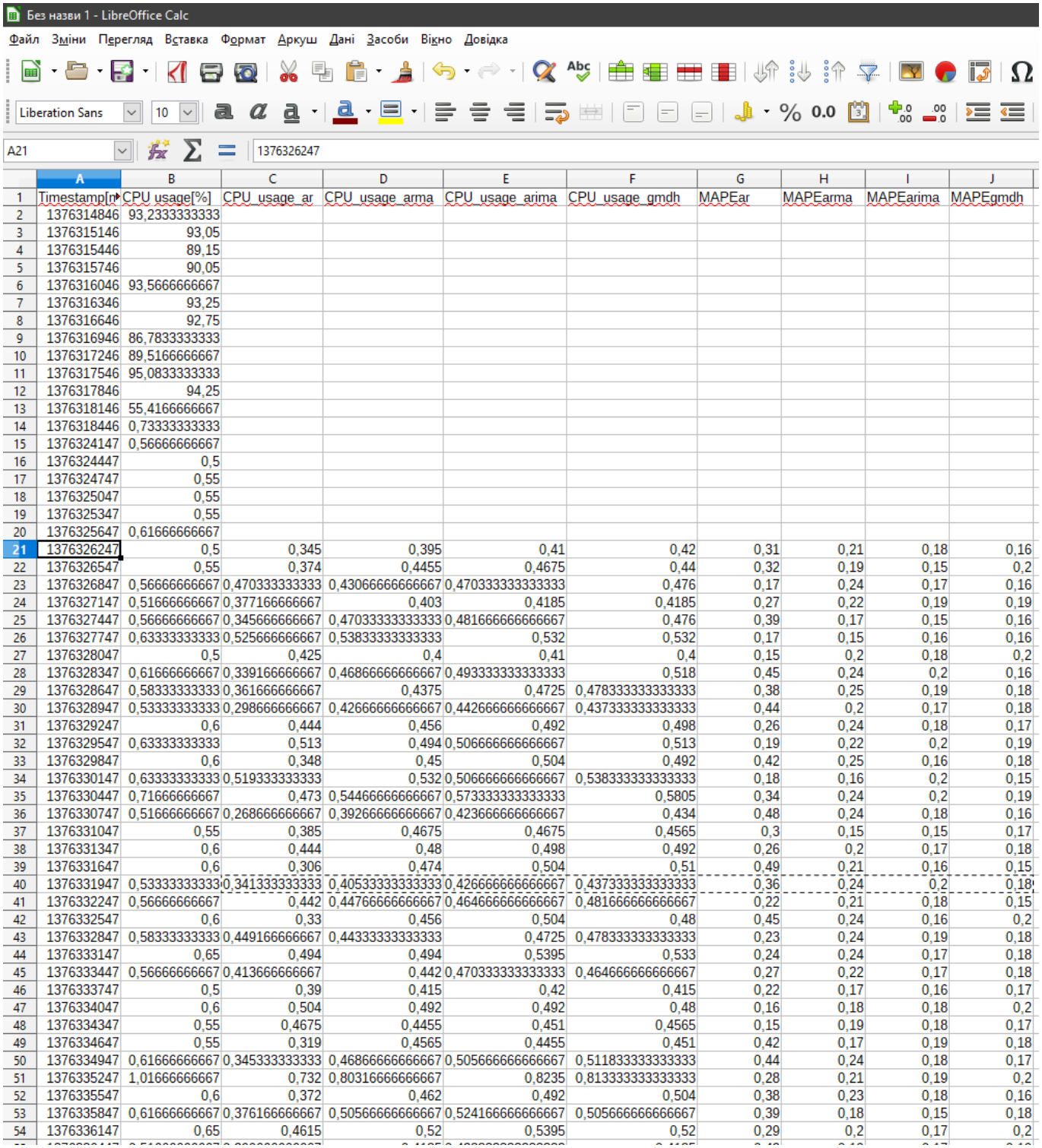
x_i – фактичне значення завантаженості CPU;

$RMSE_{i,j}$ – похибка прогнозування i-го інтервалу часу j-им алгоритмом прогнозування;

$MAPE_{i,j}$ – похибка прогнозування i-го інтервалу часу j-им алгоритмом прогнозування.

Файл містить p рядків 11 стовпців показників. Стовпці рядка розділені за допомогою "\t".

На рисунку 4.2 наведено фрагмент вмісту CSV файлу з вихідними даними.



	A	B	C	D	E	F	G	H	I	J
1	Timestamp[n	CPU usage[%]	CPU_usage_ar	CPU_usage_arma	CPU_usage_arima	CPU_usage_gmdh	MAPEar	MAPEarma	MAPEarima	MAPEgmdh
2	1376314846	93,2333333333								
3	1376315146	93,05								
4	1376315446	89,15								
5	1376315746	90,05								
6	1376316046	93,5666666667								
7	1376316346	93,25								
8	1376316646	92,75								
9	1376316946	86,7833333333								
10	1376317246	89,5166666667								
11	1376317546	95,0833333333								
12	1376317846	94,25								
13	1376318146	55,4166666667								
14	1376318446	0,73333333333								
15	1376324147	0,56666666667								
16	1376324447	0,5								
17	1376324747	0,55								
18	1376325047	0,55								
19	1376325347	0,55								
20	1376325647	0,61666666667								
21	1376326247	0,5	0,345	0,395	0,41	0,42	0,31	0,21	0,18	0,16
22	1376326547	0,55	0,374	0,4455	0,4675	0,44	0,32	0,19	0,15	0,2
23	1376326847	0,56666666667	0,470333333333	0,43066666666667	0,470333333333333	0,476	0,17	0,24	0,17	0,16
24	1376327147	0,51666666667	0,377166666667	0,403	0,4185	0,4185	0,27	0,22	0,19	0,19
25	1376327447	0,56666666667	0,345666666667	0,470333333333333	0,481666666666667	0,476	0,39	0,17	0,15	0,16
26	1376327747	0,63333333333	0,525666666667	0,53833333333333	0,532	0,532	0,17	0,15	0,16	0,16
27	1376328047	0,5	0,425	0,4	0,41	0,4	0,15	0,2	0,18	0,2
28	1376328347	0,61666666667	0,339166666667	0,46866666666667	0,493333333333333	0,518	0,45	0,24	0,2	0,16
29	1376328647	0,58333333333	0,361666666667	0,4375	0,4725	0,478333333333333	0,38	0,25	0,19	0,18
30	1376328947	0,53333333333	0,298666666667	0,42666666666667	0,442666666666667	0,437333333333333	0,44	0,2	0,17	0,18
31	1376329247	0,6	0,444	0,456	0,492	0,498	0,26	0,24	0,18	0,17
32	1376329547	0,63333333333	0,513	0,494	0,506666666666667	0,513	0,19	0,22	0,2	0,19
33	1376329847	0,6	0,348	0,45	0,504	0,492	0,42	0,25	0,16	0,18
34	1376330147	0,63333333333	0,519333333333	0,532	0,506666666666667	0,538333333333333	0,18	0,16	0,2	0,15
35	1376330447	0,71666666667	0,473	0,54466666666667	0,573333333333333	0,5805	0,34	0,24	0,2	0,19
36	1376330747	0,51666666667	0,268666666667	0,39266666666667	0,423666666666667	0,434	0,48	0,24	0,18	0,16
37	1376331047	0,55	0,385	0,4675	0,4675	0,4565	0,3	0,15	0,15	0,17
38	1376331347	0,6	0,444	0,48	0,498	0,492	0,26	0,2	0,17	0,18
39	1376331647	0,6	0,306	0,474	0,504	0,51	0,49	0,21	0,16	0,15
40	1376331947	0,53333333333	0,341333333333	0,40533333333333	0,426666666666667	0,437333333333333	0,36	0,24	0,2	0,18
41	1376332247	0,56666666667	0,442	0,44766666666667	0,464666666666667	0,481666666666667	0,22	0,21	0,18	0,15
42	1376332547	0,6	0,33	0,456	0,504	0,48	0,45	0,24	0,16	0,2
43	1376332847	0,58333333333	0,449166666667	0,44333333333333	0,4725	0,478333333333333	0,23	0,24	0,19	0,18
44	1376333147	0,65	0,494	0,494	0,5395	0,533	0,24	0,24	0,17	0,18
45	1376333447	0,56666666667	0,413666666667	0,442	0,470333333333333	0,464666666666667	0,27	0,22	0,17	0,18
46	1376333747	0,5	0,39	0,415	0,42	0,415	0,22	0,17	0,16	0,17
47	1376334047	0,6	0,504	0,492	0,492	0,48	0,16	0,18	0,18	0,2
48	1376334347	0,55	0,4675	0,4455	0,451	0,4565	0,15	0,19	0,18	0,17
49	1376334647	0,55	0,319	0,4565	0,4455	0,451	0,42	0,17	0,19	0,18
50	1376334947	0,61666666667	0,345333333333	0,46866666666667	0,505666666666667	0,511833333333333	0,44	0,24	0,18	0,17
51	1376335247	1,01666666667	0,732	0,80316666666667	0,8235	0,813333333333333	0,28	0,21	0,19	0,2
52	1376335547	0,6	0,372	0,462	0,492	0,504	0,38	0,23	0,18	0,16
53	1376335847	0,61666666667	0,376166666667	0,50566666666667	0,524166666666667	0,505666666666667	0,39	0,18	0,15	0,18
54	1376336147	0,65	0,4615	0,52	0,5395	0,52	0,29	0,2	0,17	0,2

Рисунок 4.2 – Фрагмент вмісту CSV файлу з вихідними даними

4.2 Програмне та технічне забезпечення

4.2.1 Засоби розробки

Для програмної реалізації алгоритму прогнозування потреби ресурсів для серверної системи в умовах хмарного обчислення використано мову C# [48], програмне середовище R для статистичних обчислень, аналізу та зображення даних в графічному вигляді та інтегроване середовище програмування Visual Studio [49].

R — мова програмування [50] і програмне середовище для статистичних обчислень, аналізу та відображення даних в графічному вигляді. Розробка R відбувалась під істотним впливом двох наявних мов програмування: мови програмування S з семантикою успадкованою від Scheme. R названа за першою літерою імен її засновників Роса Іхаки та Роберта Джентлмена працівників Оклендського Університету в Новій Зеландії. Незважаючи на деякі принципові відмінності, більшість програм, написаних мовою програмування S запускаються в середовищі R.

R має значні можливості для здійснення статистичних аналізів, включаючи лінійну і нелінійну регресію, класичні статистичні тести, аналіз часових рядів (серій), кластерний аналіз і багато іншого. R легко розбудовується завдяки використанню додаткових функцій і пакетів, доступних на сайті Comprehensive R Archive Network (CRAN). Більша частина стандартних функцій R написана мовою R, однак існує можливість підключати код написаний на C, C++, або Фортран. Також, за допомогою програмного коду на C# або Java, можна безпосередньо маніпулювати R об'єктами.

R підтримує концепцію об'єктно-орієнтованого програмування (ООП), включаючи generic функції, результат виконання якої залежить від аргументів (типу об'єктів), що передаються generic функції. В мові програмування R всі змінні є об'єктами, кожен об'єкт належить до певного класу. При цьому R має дві класові моделі: S3 та S4. Перша була реалізована від початку існування R, друга була додана у версії 1.7.0 з пакетом methods.

Можливості R значно розширюються додатковими пакетами (бібліотеками). Пакети розробляються безпосередньо користувачами R. Існує понад 4500 пакетів, доступних на сайті Comprehensive R Archive Network (CRAN), Omegahat, Bioconductor, R-Forge.

Лінійка продуктів компанії Microsoft, що включають інтегроване середовище розробки програмного забезпечення і ряд інших інструментальних засобів. Дані продукти дозволяють розробляти як консольні додатки, так і додатки з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-додатки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, підтримуваних Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework і Silverlight.

Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю найпростішого рефакторінга коду. Вбудований відладчик може працювати як відладчик рівня вихідного коду, так і відладчик машинного рівня. Решта вбудовуються інструменти включають в себе редактор форм для спрощення створення графічного інтерфейсу додатку, веб-редактор, дизайнер класів і дизайнер схеми бази даних. Visual Studio дозволяє створювати і підключати сторонні додатки (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (як, наприклад, Subversion і Visual SourceSafe), додавання нових наборів інструментів (наприклад, для редагування і візуального проектування коду на предметно-орієнтованих мовах програмування) або інструментів для інших аспектів процесу розробки програмного забезпечення (наприклад, клієнт Team Explorer для роботи з Team Foundation Server). Visual Studio підтримує інструменти (у вигляді плагінів) для проведення тестування, системи контролю версій (наприклад, Git).

Для виклику скриптів R через мову програмування C#, нам знадобиться бібліотека R.Net її можна встановити за допомогою NuGet [51]. NuGet - ключовий інструмент для будь-якої сучасної платформи розробки - це механізм, за допомогою якого розробники можуть створювати, передавати один одному і використовувати

код. Часто такий код розподілений по "пакетах", що включає скомпільований код (у вигляді бібліотек DLL) і інший вміст, необхідний для використання цих пакетів проектами.

Для .NET (в тому числі .NET Core) механізмом спільного використання коду, підтримуваним Майкрософт, є NuGet, який визначає, як створюються, розміщуються і використовуються пакети для .NET, а також надає засоби для кожної з цих ролей.

Простіше кажучи, пакет NuGet є окремим ZIP-файлом з розширенням .nupkg, який містить скомпільований код (DLL), інші файли, пов'язані з цим кодом, і описовий маніфест, що включає такі відомості, як номер версії пакета. Розробники, у яких є код, до якого потрібно надати загальний доступ, створюють пакети і публікують їх на закритих або відкритих вузлах. Споживачі отримують ці пакети з відповідних вузлів, додають їх у свої проекти, а потім викликають функції пакета в коді свого проекту. При цьому NuGet сам обробляє всі проміжні дані.

Так як NuGet підтримує закриті вузли поряд з відкритим вузлом nuget.org, за допомогою пакетів NuGet є можливість ділитися кодом, використовуваним в рамках організації або робочої групи. Пакети NuGet також є зручним способом факторізовать свій код для використання тільки у власних проектах. Іншими словами, пакет NuGet є спільно використовуваною одиницею коду, проте не вимагає і не має на увазі будь-якого певного способу надання загального доступу.

Для того, щоб почати використовувати в C# коди з R, потрібно ініціалізувати (рисунок 4.3) змінні системного середовища (англ. system environment variables) та створити екземпляр об'єкту Regine.

```
string rHome = @"C:\Program Files\R\R-3.4.4";
string rPath = @"C:\Program Files\R\R-3.4.4\bin\i386";
REngine.SetEnvironmentVariables(rPath, rHome);
REngine engine = REngine.GetInstance();
engine.Initialize();
```

Рисунок 4.3– Ініціалізація змінних системного середовища та створення екземпляр об'єкта Regine.

Після цього, можна починати працювати з кодом R. Щоб викликати команди R в C# використовують метод Evaluate (рисунок 4.4) для виконання команд R.

```
engine.Evaluate("vm <- read.csv('E:/fastStorage/2013-8/new/1.csv'
engine.Evaluate("y = vm$CPUusageP");
engine.Evaluate("p=4");
engine.Evaluate("modelAR <- arima(y, order = c(p, 0, 0))");
```

Рисунок 4.4 – Приклад викликів команд R в мові програмування C#.

4.2.2 Вимоги до технічного забезпечення

4.2.2.1 Загальні вимоги

Технічні характеристики робочих комп'ютерів користувачів повинні відповідати вимогам до середовища виконання .Net framework та програмне середовище R, які повинні бути інстальовані на робочому комп'ютері користувача з операційною системою Microsoft Windows Vista, 7, 8.

4.2.3 Архітектура програмного забезпечення

4.2.3.1 Діаграма класів

У частині графічного матеріалу представлена схема структурна класів ПЗ, які відповідають за виконання таких функцій, як розбір CSV файлу з завантаженістю ВМ, збереження даних процесу прогнозування, збереження вихідних даних в CSV файл, запуск прогнозування потреби ресурсів за допомогою методів AR, ARMA, ARIMA, GMDH та гібрідного, відображення графічного інтерфейсу користувача, візуалізації отриманих результатів прогнозування.

Схема містить 13 класів, а саме:

– Main – клас, що реалізує графічний інтерфейс користувача та візуалізацію отриманих результатів прогнозування;

- ParseCSV – клас, що відповідає за розбір CSV файлу з показниками завантаженості CPU;
- WriteToCSV – клас, що відповідає за запис даних в csv файл;
- ARIMA – клас, що реалізує прогнозування за допомогою моделі ARIMA;
- AR – клас, що реалізує прогнозування за допомогою моделі AR;
- ARMA – клас, що реалізує прогнозування за допомогою моделі ARMA;
- GMDH – клас, що реалізує прогнозування за допомогою моделі GMDH;
- HYBRID – клас, що реалізує прогнозування за допомогою гібрідного алгоритму;
- OpenFileDialog – показує діалогове вікно, що дозволяє користувачеві відкрити файл;
- SaveFileDialog - пропонує користувачеві ввести місце для збереження файлу.
- StreamReader – реалізує об'єкт TextReader, який зчитує символи з потоку байтів в певному кодуванні;
- StreamWriter – реалізує TextWriter для запису символів в потік в певному кодуванні.
- Chart – клас, що виводить графіки.

4.2.3.2 Специфікація функцій

Специфікація функцій наведена в таблиці 4.1.

Таблиця 4.1 – Специфікація функцій

Клас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
SaveFileDialog	CreateObjRef	Створює об'єкт, який містить всі необхідні відомості, необхідні	ObjRef	-	

		для створення проксі-сервері, який використовується для обміну даними з віддаленим об'єктом.			
SaveFileDialog	Dispose	Звільняє всі ресурси	-	-	-

Продовження таблиці 4.1

Клас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
SaveFileDialog	Equals	Визначає, чи рівний заданий об'єкт поточного об'єкту	boolean	-	-
SaveFileDialog	GetHashCode	Служить хеш-функцією за замовчуванням	String	-	-
SaveFileDialog	OpenFile	Відкриває файл з дозволами читання і запису, вибраної	-	-	-

		користувачем			
SaveFileDialog	Reset	Повертає всі параметрами діалогового вікна значення за замовчуванням	-	-	-
SaveFileDialog	ShowDialog	Запускає загальне діалогове вікно з заданим за замовчуванням власником	-	-	-

Продовження таблиці 4.1

Клас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
OpenFileDialog	CreateObjRef	Створює об'єкт, який містить всі необхідні відомості, необхідні для створення проксі-серверів, який використовується для обміну даними з	ObjRef	-	-

		віддаленим об'єктом.			
OpenFileDialog	Dispose	Звільняє всі ресурси	-	-	-
OpenFileDialog	Equals	Визначає, чи рівний заданий об'єкт поточного об'єкту	boolean	-	-
OpenFileDialog	GetHashCode	Служить хеш-функцією за замовчуванням	String	-	-
OpenFileDialog	OpenFile	Відкриває файл з дозволами читання і запису, вибраної користувачем	-	-	-

Продовження таблиці 4.1

Клас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
OpenFileDialog	Reset	Повертає всі параметрами діалогового вікна значення за замовчуванням	-	-	-
OpenFileDialog	ShowDialog	Запускає загальне діалогове вікно з заданим за	-		-

		замовчуванням власником			
StreamReader	Close	Закриває об'єкт StreamReader і основний потік і звільняє всі системні ресурси, пов'язані з пристроєм читання.	-	-	-

Продовження таблиці 4.1

Клас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
StreamReader	DiscardBufferedData	Очищає внутрішній буфер		-	-
StreamReader	Dispose	Звільняє всі ресурси, використовувані об'єктом TextReader		-	-
StreamReader	Finalize	Дозволяє об'єкту спробувати звільнити ресурси і виконати інші операції очищення, перед тим як він буде знищений під час збирання сміття		-	-
StreamReader	Peek	Повертає наступний доступний символ,	char	-	-

Продовження таблиці 4.1

Клас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
StreamReader	Read	Виконує читання наступного символу з вхідного потоку і переміщує стан символу на одну позицію вперед	char	-	-
StreamReader	ReadLine	Виконує читання рядка символів з поточного потоку і повертає дані у вигляді рядка	string	-	-
StreamReader	ReadToEnd	Зчитує всі символи, починаючи з поточної позиції до кінця потоку	string	-	-

Продовження таблиці 4.1

Клас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
StreamWriter	Close	Закриває поточний StreamWriter об'єкта і основний потік.	-	-	-
StreamWriter	Dispose	Звільняє всі ресурси, використовувані об'єктом TextWriter	-	-	-
StreamWriter	Finalize	Дозволяє об'єкту спробувати звільнити ресурси і виконати інші операції очищення	-	-	-
StreamWriter	Flush	Очищає всі буфери для поточного засоби запису та викликає запис всіх даних буфера	-	-	-

Продовження таблиці 4.1

аКлас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
StreamWriter	Write	Записує в потік рядок	-	-	Результати прогнозування
StreamWriter	WriteLine	Записує ознака кінця рядка в текстовий рядок або потік	-	-	Результати прогнозування
AR	Create Model	Створює модель AR	Model	-	Коефіцієнт моделі AR
AR	Forecast	Функція для прогнозування з часових рядів або моделей часових рядів	Double[]	-	
AR	GetTimeSeries	Отримати часовий ряд	Double[]	-	
AR	SetTimeSeries	Задати часовий ряд для моделі	-	Double[]	Дані часового ряду
ARMA	Create Model	Створює модель ARMA	Model	-	-

Продовження таблиці 4.1

Клас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
ARMA	Forecast	Функція для прогнозування з часових рядів або моделей часових рядів	Double[]	-	
ARMA	GetTimeSeries	Отримати часовий ряд	Double[]	-	Отримання часового ряду
ARMA	SetTimeSeries	Задати часовий ряд для моделі	-	Double[]	Дані часового ряду
ARIMA	CreateModel	Створює модель ARIMA	-	-	-
ARIMA	Forecast	Функція для прогнозування часових рядів, моделей часових рядів	-	-	-
ARIMA	GetTimeSeries	Отримати часовий ряд	Double[]	-	

Продовження таблиці 4.1

Клас	Метод	Призначення	Повертає результат	Список параметрів	Семантика параметрів
ARIMA	SetTimeSeries	Задати часовий ряд для моделі	-	Double[]	Дані часового ряду
GMDH	Forecast	прогнозує часові ряди за допомогою алгоритмів нейронних мереж типу GMDH	Double[]	-	-
GMDH	GetTimeSeries	Отримати часовий ряд	Double[]	-	
GMDH	SetTimeSeries	Задати часовий ряд для моделі	-	Double[]	Дані часового ряду

4.2.4 Керівництво користувача

Перед початком роботи з ПЗ необхідно інсталювати програмне середовище R в бажану директорію.

Для її інсталяції необхідно перейти за посиланням https://cran.r-project.org/bin/, обрати останню або необхідну користувачу версію R (рисунок 4.4) та в обраній версії натиснути на посилання «Download» (рисунок 4.4)

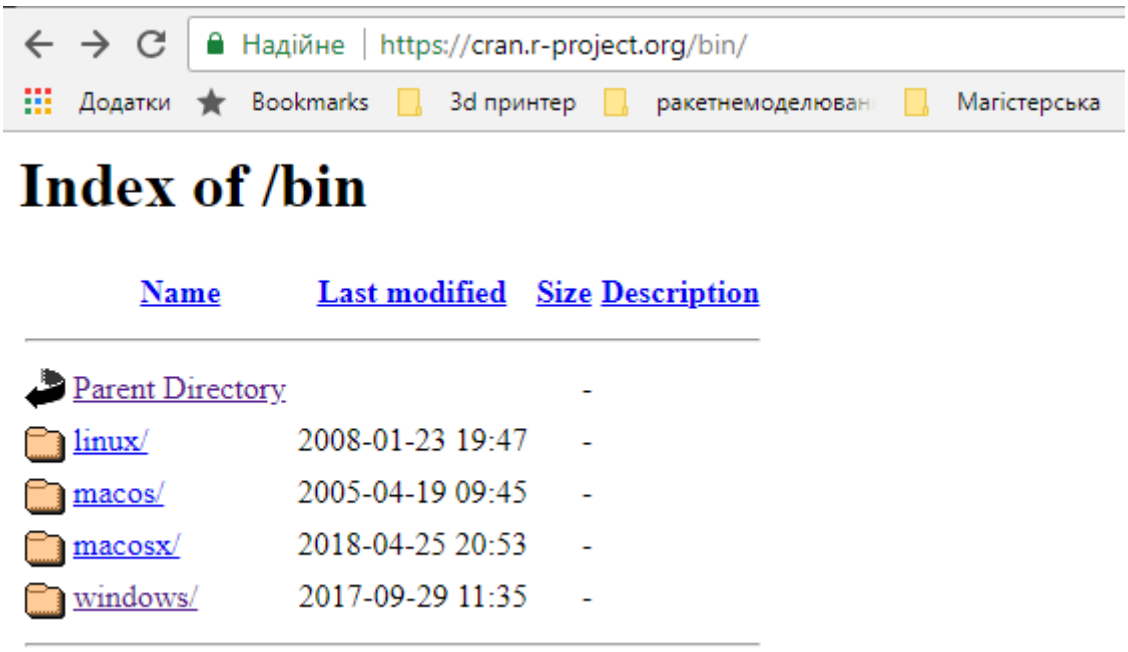


Рисунок 4.3– Вибір R для ОС

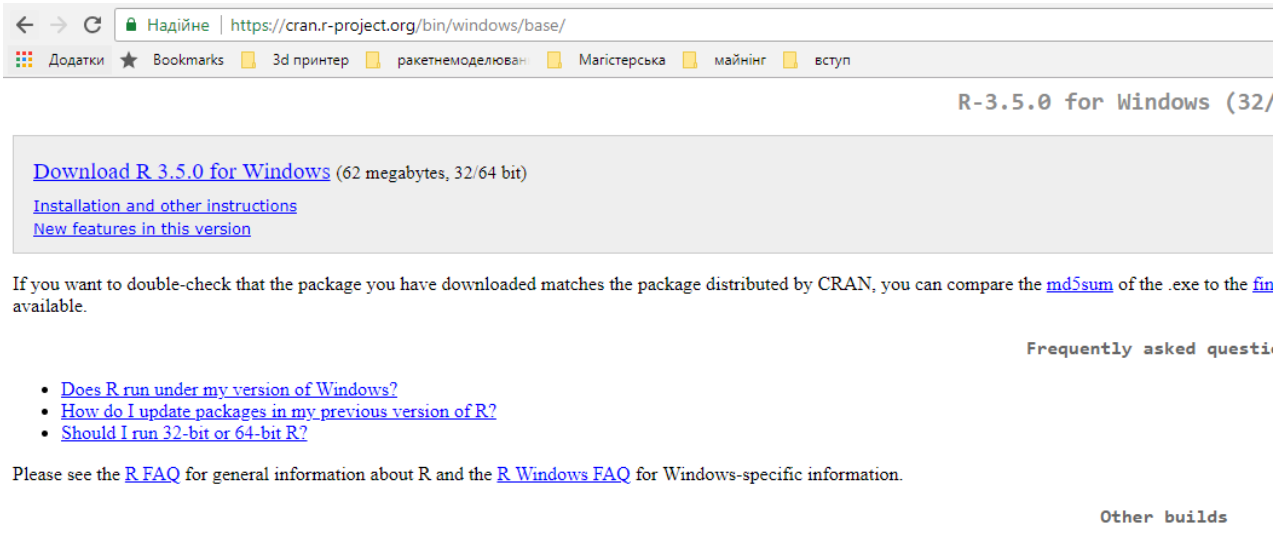


Рисунок 4.4 – Завантаження програмного середовища R

Після закінчення завантаження, необхідно відкрити збережений файл (рисунок 4.5) та натиснути на кнопку «Install».

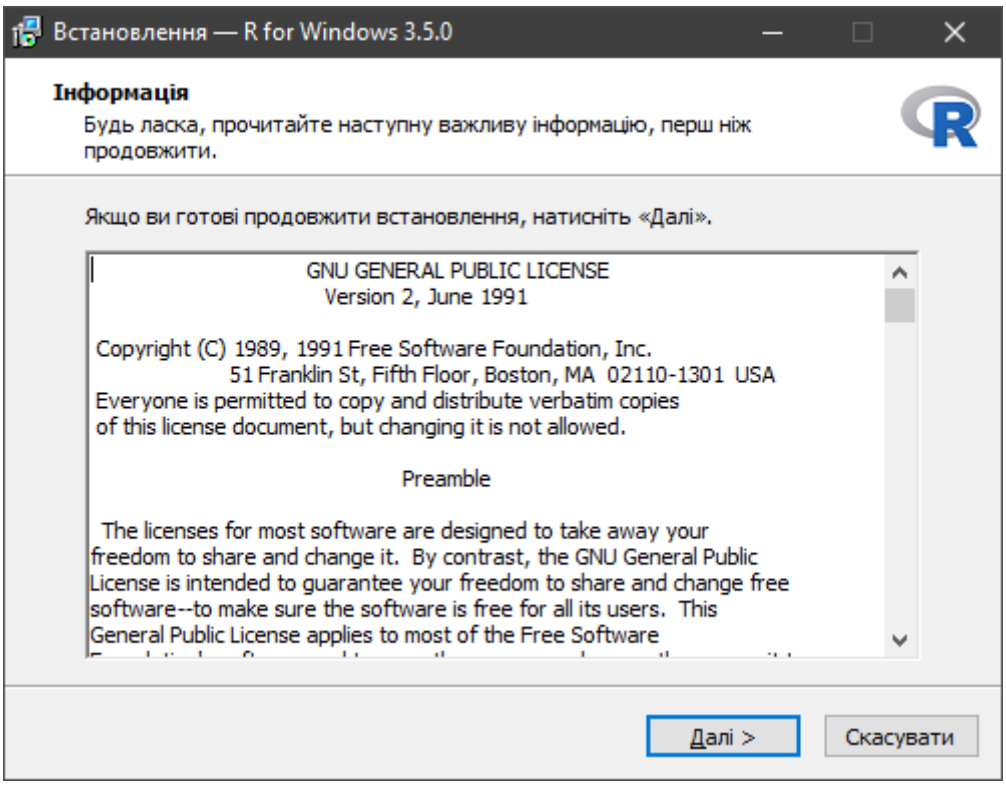


Рисунок 4.5 – Форма інсталяції програмного середовища R

Обрати категорію інсталяції та натиснути далі (рисунок 4.6).

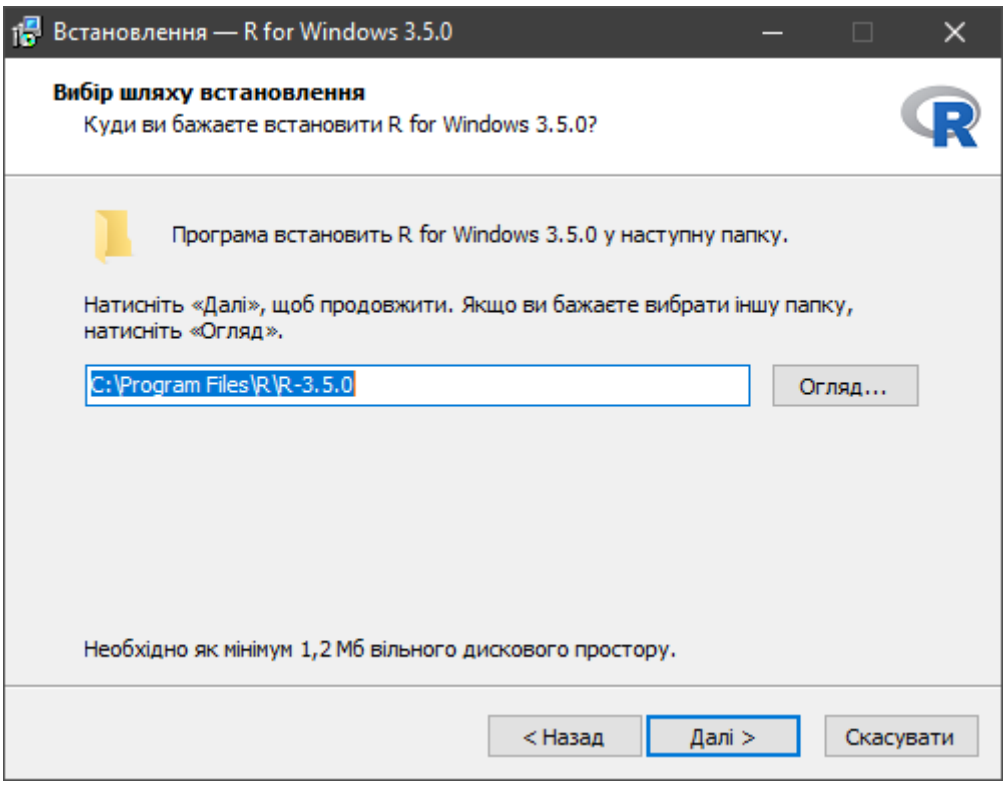


Рисунок 4.6 – Форма вибору категорії

Після форми вибору категорії, буде запропоновано обрати обрати розрядність програмного забезпечення та додаткові компоненти.(рисунок 4.7)

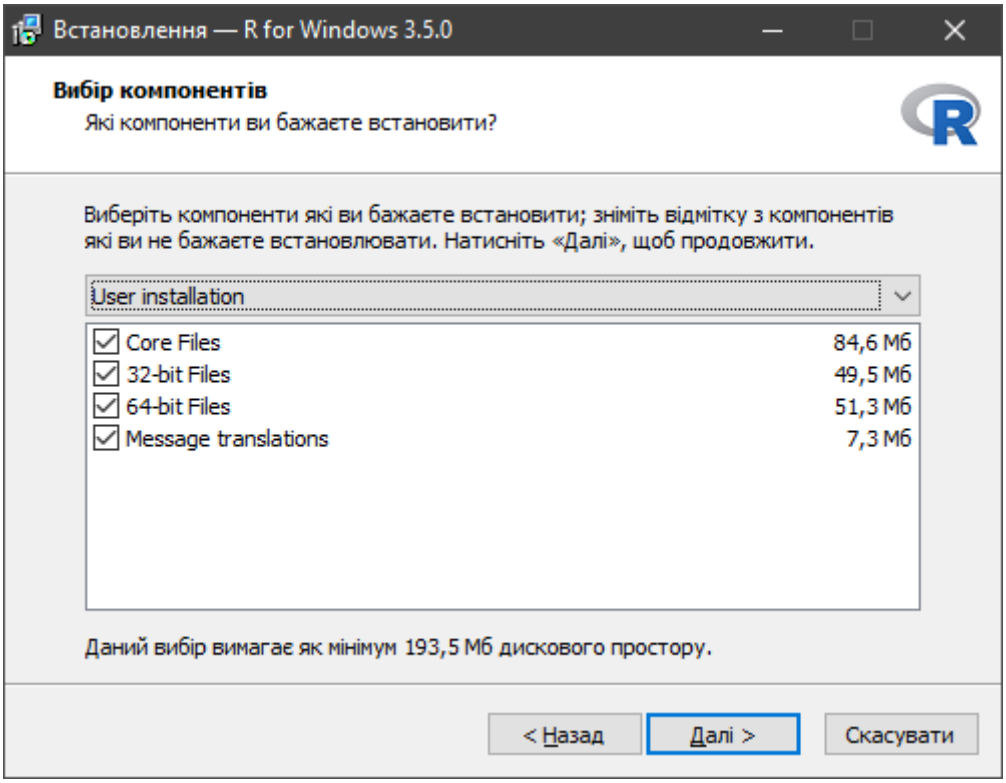


Рисунок 4.7 – Вибір додаткових компонентів програмного середовища

Після вибору всіх налаштувань, почнеться інсталіцяв і на кінцевому етапі відкриється форма завершення інсталяції (рисунок 4.8).

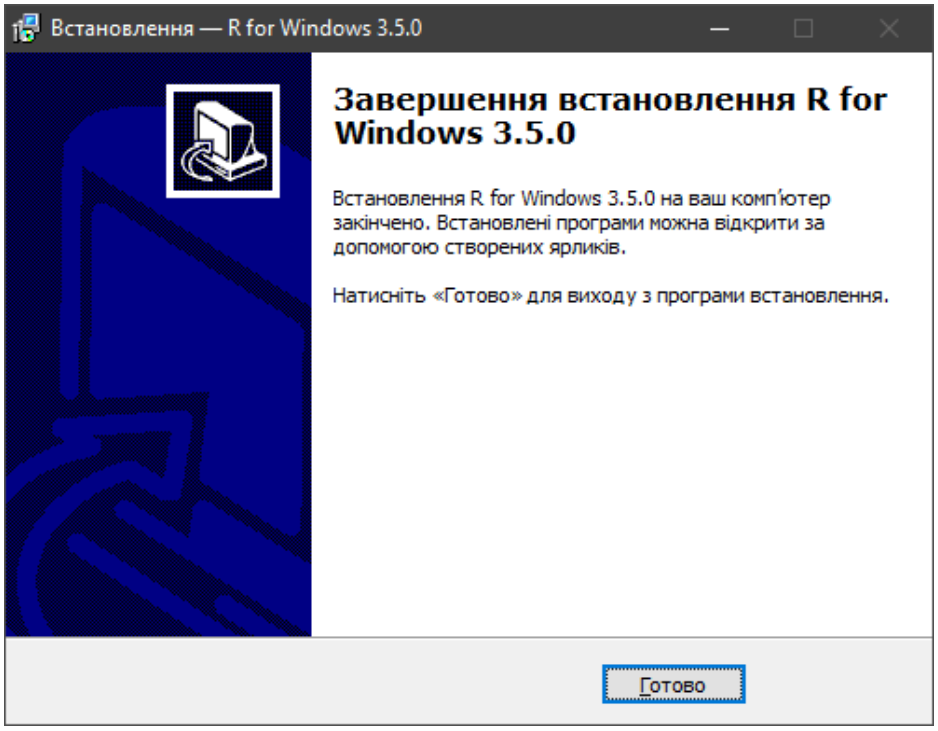


Рисунок 4.8 – Вибір додаткових компонентів програмного середовища

Наступним кроком буде установка розробленого програмного забезпечення. Щоб установити програмне забезпечення, запустити файл ForecastSetup.exe (рисунок 4.9).

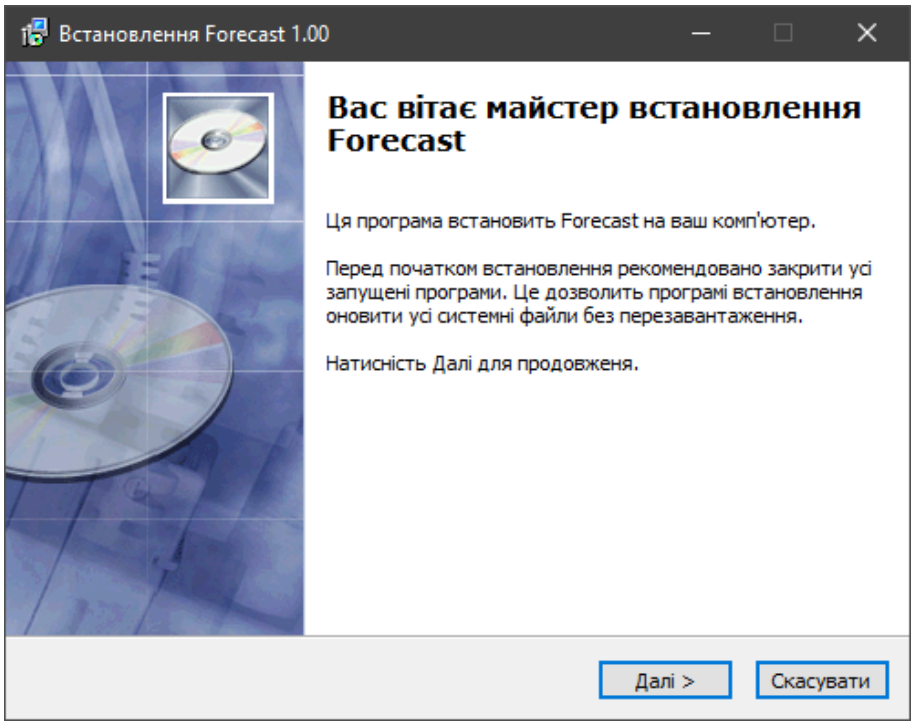


Рисунок 4.9 – Запуск інсталяції розробленого програмного забезпечення

Після запуску потрібно обрати категорію інсталяції (рисунок 4.10) натустинути далі та дочекатись завершення інсталяції (рисунок 4.11)

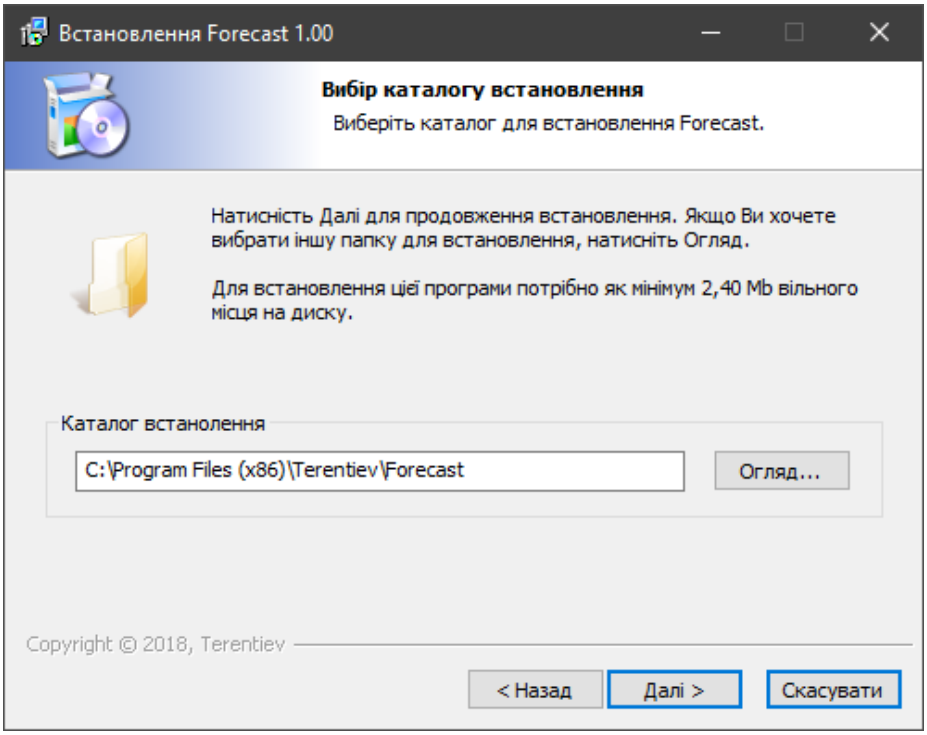


Рисунок 4.10 – Форма вибору категорії

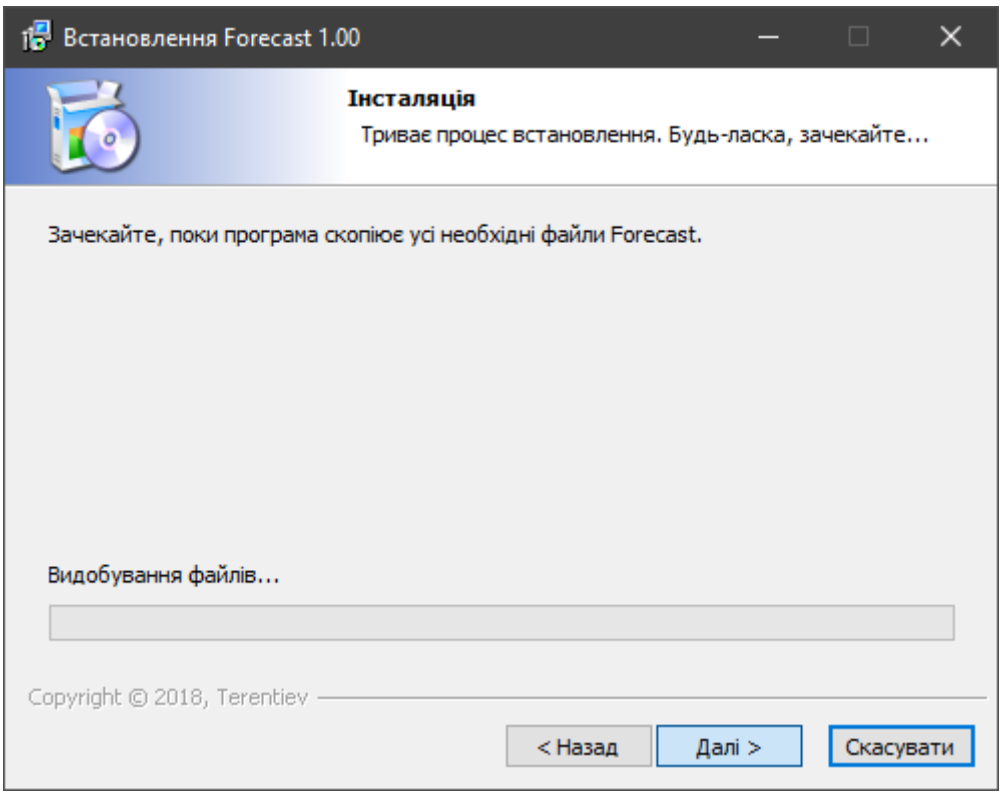


Рисунок 4.11 – Завершення інсталяції

Для запуску ПЗ необхідно запустити на робочому столі файл Forecast.exe та відкриється головне вікно ПЗ (рисунок 4.12)

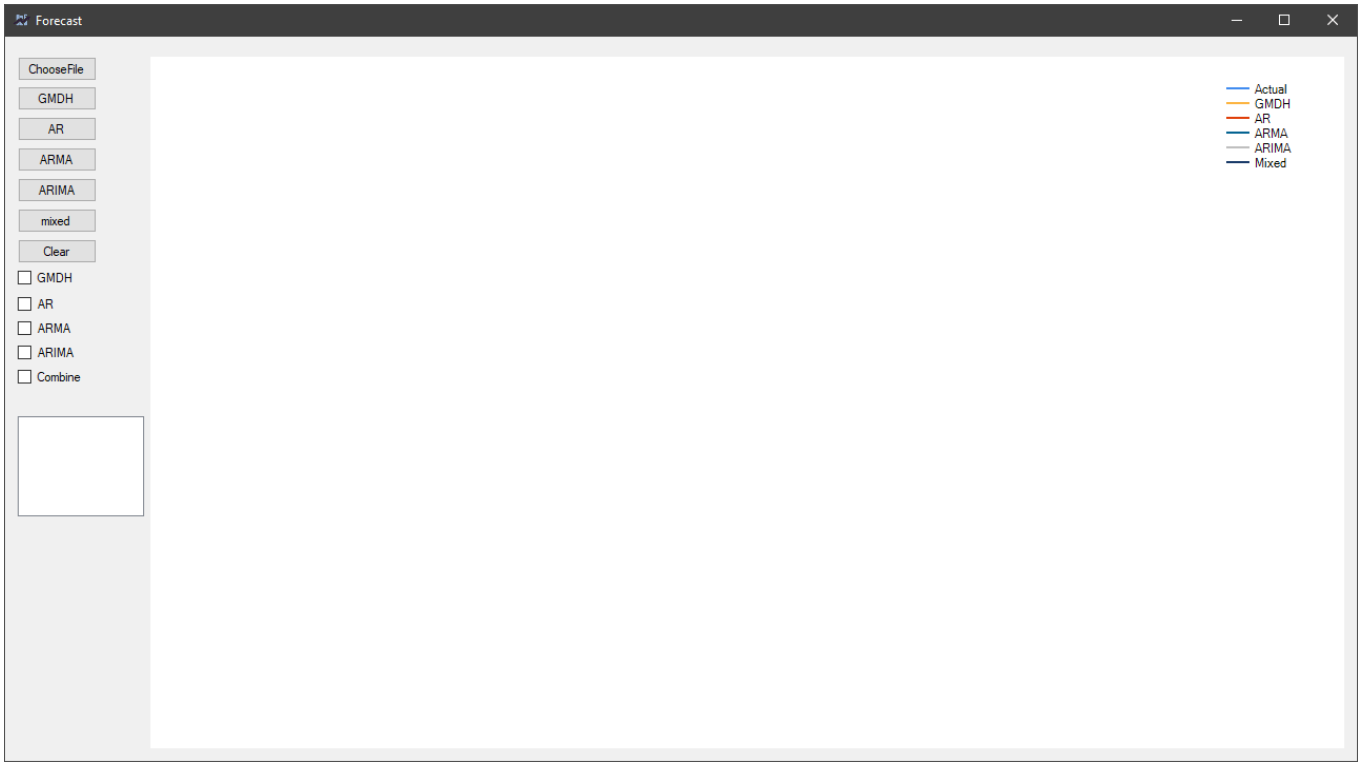


Рисунок 4.12 – Головне вікно ПЗ

Для завантаження CSV файлу з завантаженням процесора потрібно натиснути кнопку «ChooseFile» у головному вікні ПЗ (рисунок 4.13).

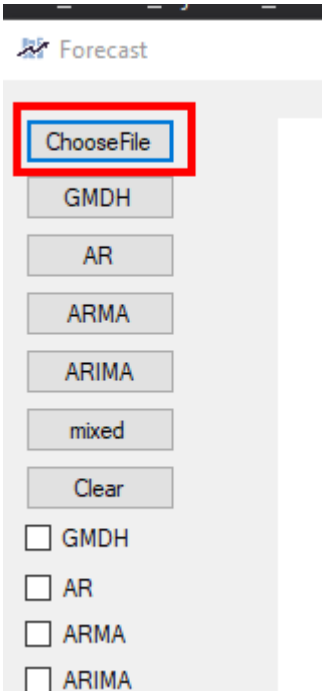


Рисунок 4.13 – Кнопка «ChooseFile»

Після цього відкриється діалогове вікно для вибору CSV файлу (рисунок 4.14).

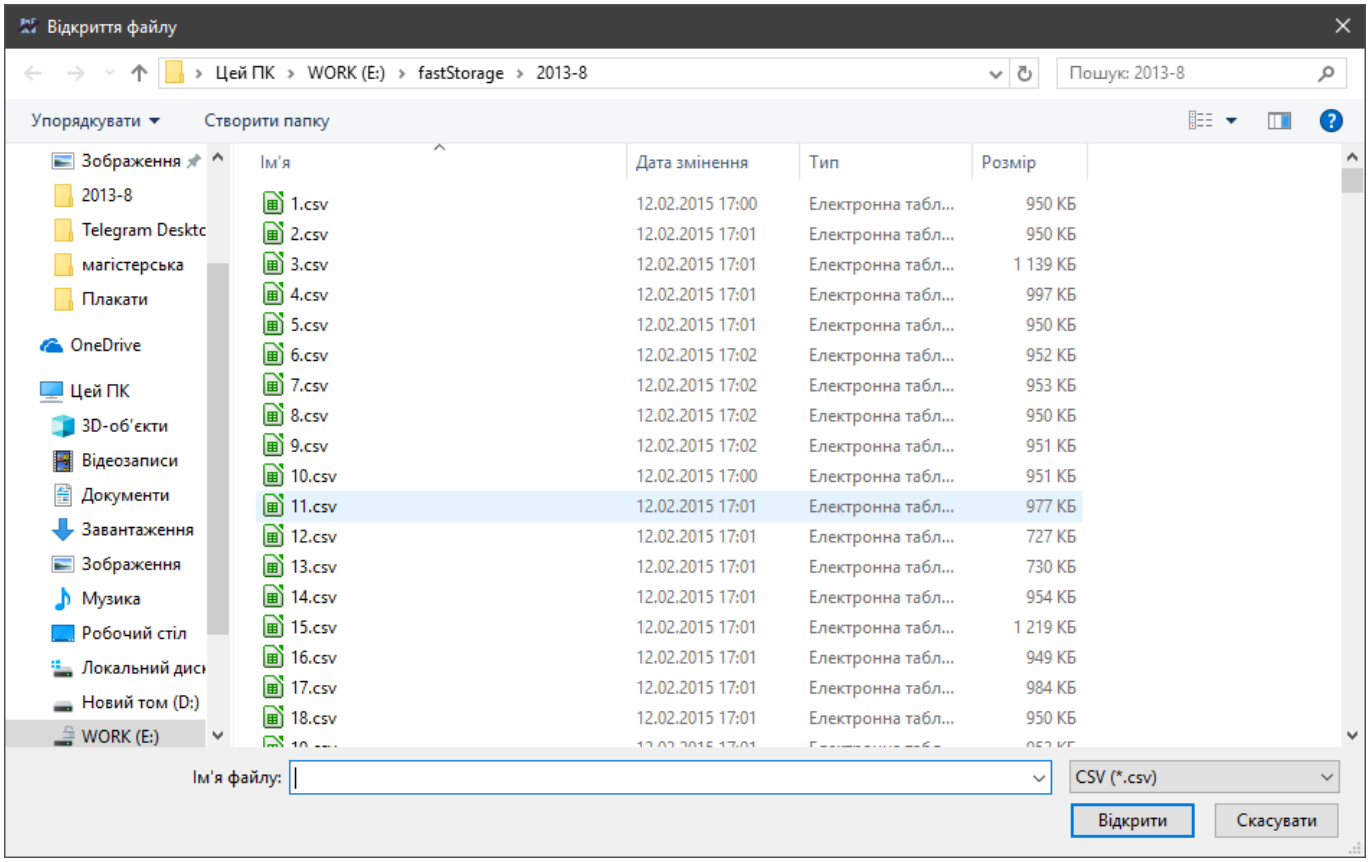


Рисунок 4.14 – Діалогове вікно для вибору CSV файлу

Потім у списку потрібно обрати стовпець по якому будемо робити прогнозування(рисунок 4.15) та запустити прогнозування потреби процесора для серверноъ системи, натиснувши на одну з кнопок «GMDH, AR, ARMA, ARIMA, mixed» (рисунок 4.16).

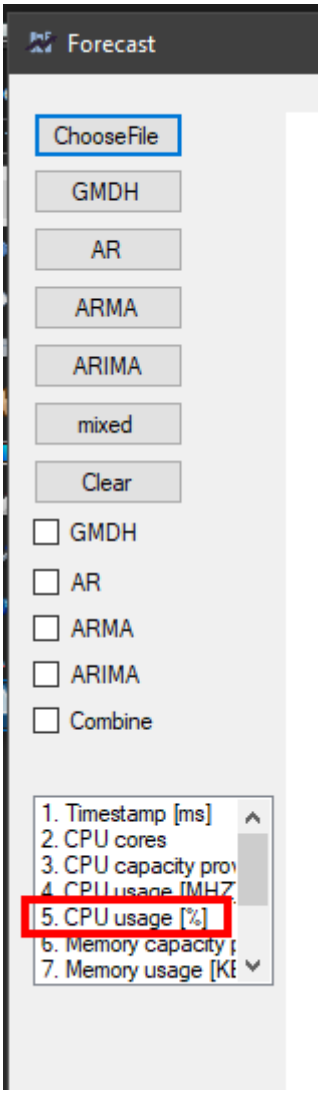


Рисунок 4.15 – Список полів, які доступні для прогнозування в CSV файлів

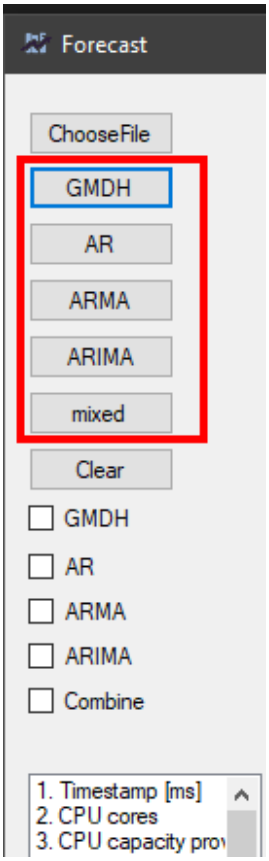


Рисунок 4.16 – Запуск прогнозування

Після прогнозування можливо очистити результати, для цього потрібно натиснути кнопку «Clear» (рисунок 4.17).

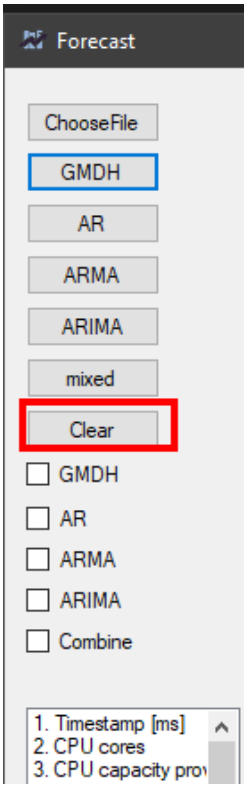


Рисунок 4.17 – Очищення результатів прогнозування

Після завершення прогнозування у головному вікні ПЗ відображені графіки прогнозування завантаженості процесора обраними алгоритмами та фактичне завантаження процесора (рисунок 4.).

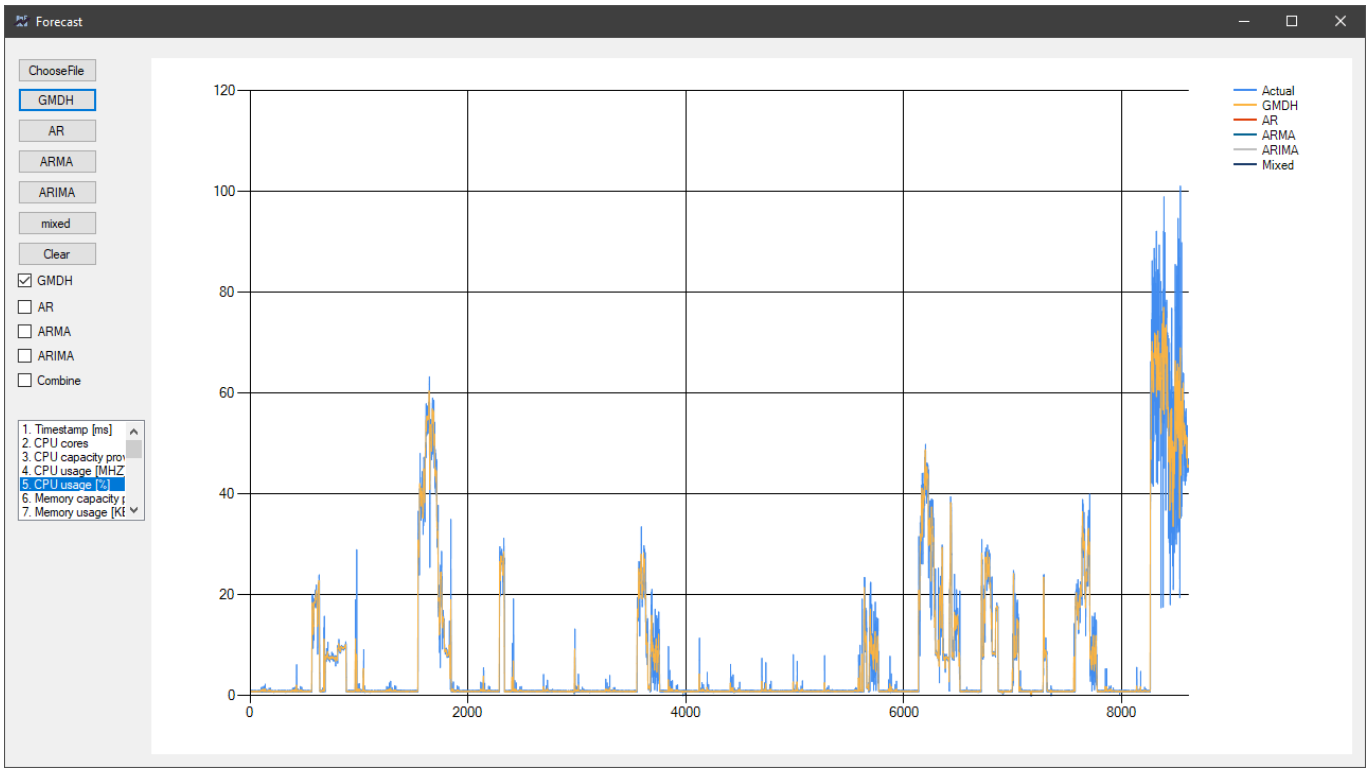


Рисунок 4.18 – Відображення результатів прогнозування завантаженості серверної системи в умовах хмарного обчислення

За допомогою прапорців (рисунок 4.19) можна вмикати та вимикати графіки різних алгоритмів прогнозування.

Рисунок 4.19 – Вмикання та вимкнення графіків прогнозування

У директорії outputs створено файл з розширенням .csv, у якому містяться вихідні дані (опис формату файлів з вихідними даними дивитися в пункті 4.1.2).

У разі виникнення помилки під час роботи ПЗ відкриється вікно з повідомленням про помилку (рисунок 4.20).

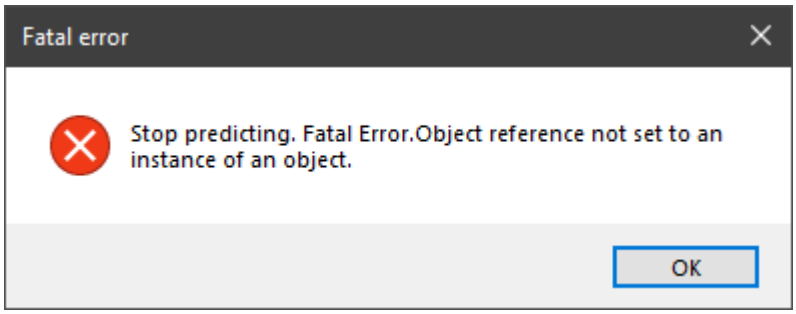


Рисунок 4.20 – Повідомлення про помилку

4.3 Висновок до розділу

У даному розділі наведений опис форматів вхідних і вихідних даних. Наведений опис полів, які містяться у CSV файлі з характеристики ВМ , приклад вмісту CSV файлу.

Детально описані переваги та недоліки засобів розробки алгоритму прогнозування потреби ресурсів для серверної системи в умовах хмарного обчислення на основі AR, ARMA, ARIMA, GMDH.

Наведена схема структурна класів ПЗ, а також специфікацію функцій. Наведено детальне керівництво користувача.

5 РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ

5.1 Порядок проведення досліджень

На початку проведення досліджень було завантажено вхідні дані від компанії bitbrains [52] з завантаженістю ВМ. Збір вхідних даних виконано на 1250 ВМ. Отримані результати використані при роботі з розробленим програмним продуктом.

При роботі віртуальної машини, її завантаженість постійно коливається, а більшість ВМ незавантажується навіть на 40%, тому необхідно постійно мати актуальний прогноз, щоб можна було передавати на апарат управління сигнали про консолідацію ВМ на серверній системі.

Прогнозування завантаженості CPU ВМ, методами AR (рисунок 5.1), ARMA (рисунок 5.2), ARIMA (рисунок 5.3), GMDH (рисунок 5.4) та розробленим гібрідним (рисунок 5.5) методом будуть проводитись на реальних даних 1250 ВМ зібраних з інтервалами в 5 хвилин з розподіленого ЦОД Bitbrains протягом одного місяця.

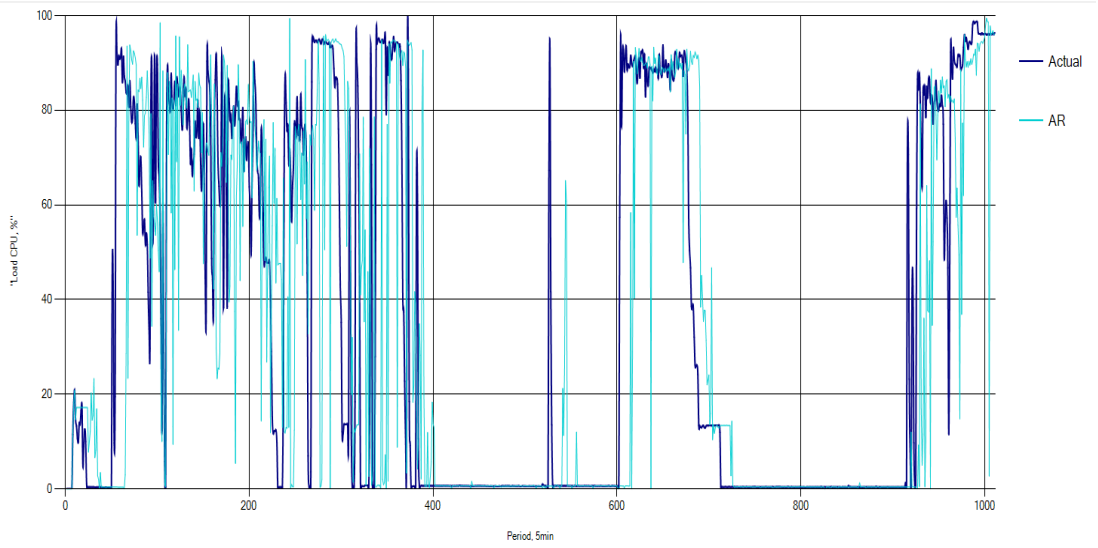


Рисунок 5.1 – Прогнозування методом AR

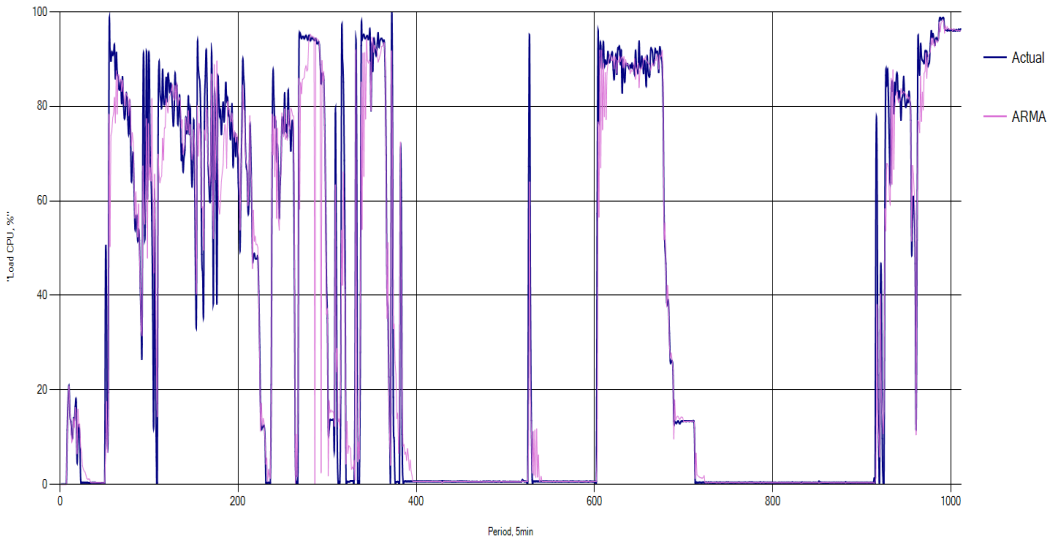


Рисунок 5.2 – Прогнозування методом ARMA

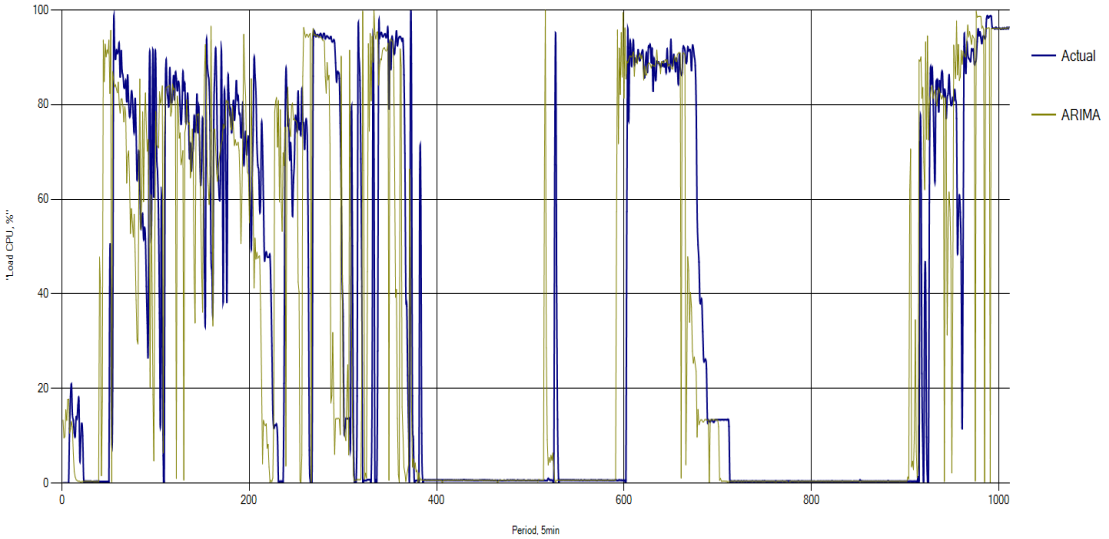


Рисунок 5.3 – Прогнозування методом ARIMA

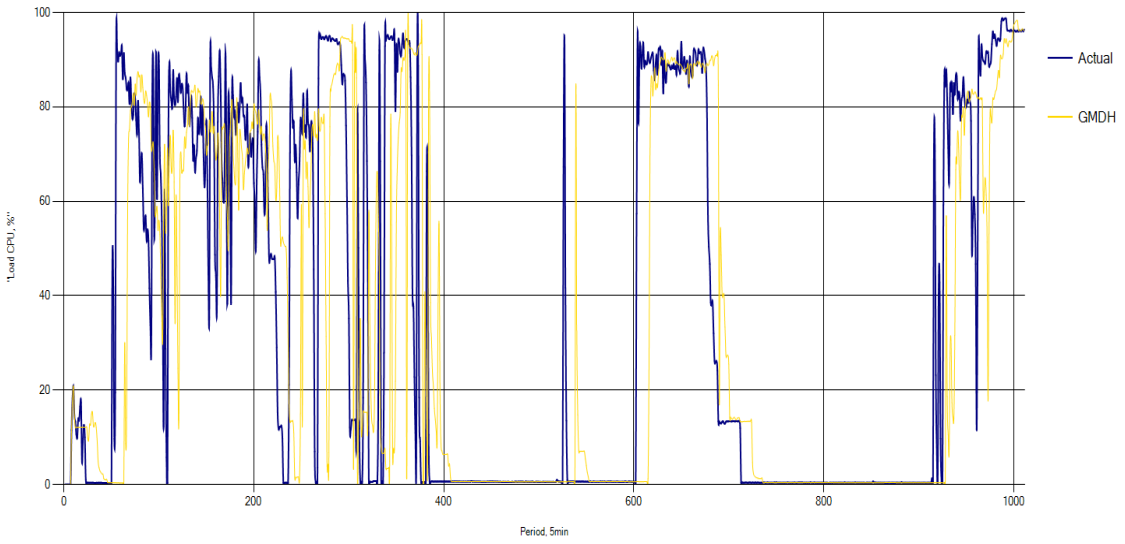


Рисунок 5.4 – Прогнозування методом GMDH

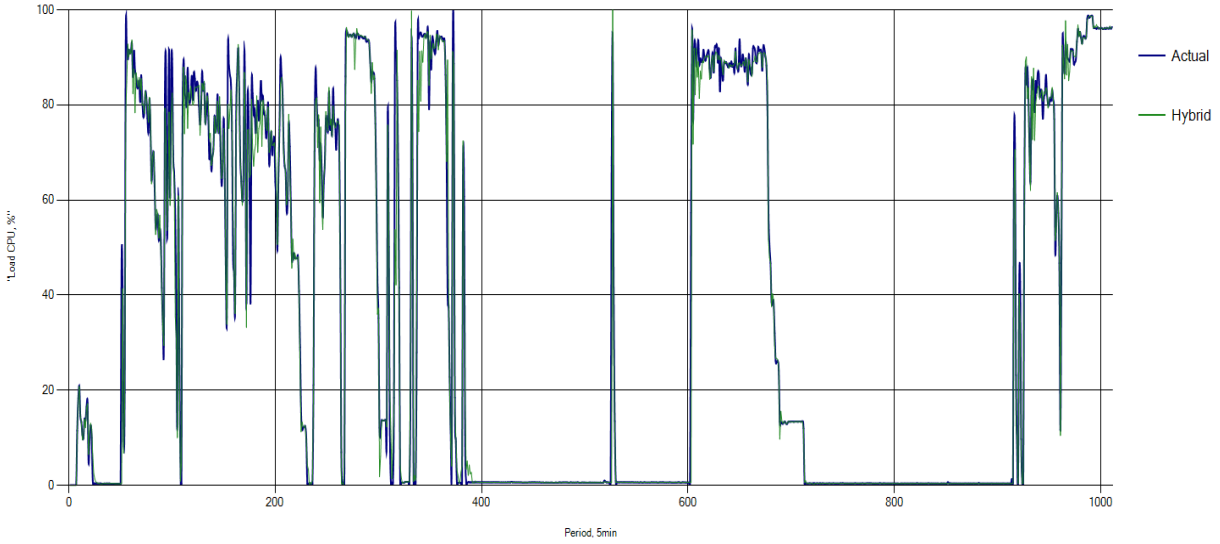


Рисунок 5.5 – Прогнозування гібрідним методом

Для кожного методу було розраховано середнє квадратичне відхилення, на основі всіх прогнозувань, отримані результати наведені у таблиці 5.1

Таблиця 5.1 – Середньо квадратичні відхилення кожним методом прогнозування

$mean(RMSE_{AR})$	$mean(RMSE_{ARMA})$	$mean(RMSE_{ARIMA})$	$mean(RMSE_{GMDH})$	$mean(RMSE_{Hybrid})$
47,12%	43,72	35,733%	30,64%	18,18%

5.2 Висновки до розділу

В представленому розділі було наведено порядок проведення досліджень, визначені вхідні дані, проаналізовано отримані результати використання алгоритмів прогнозування завантаженості центрального процесора. Також, виконано порівняння роботи прогнозування потреби ресурсів представлених алгоритмів та гібрідного алгоритму.

Визначено, що за результатами прогнозування гібрідним алгоритмом, є ефективнішим ніж інші методи прогнозування, проаналізовані в даній магістерській дисертації.

ЗАГАЛЬНІ ВИСНОВКИ

Під час виконання магістерської дисертаціїї розглянуто поширені методи прогнозування часових рядів, проведено їх порівняльний аналіз та визначення їх ефективності при прогнозуванні потреби ресурсів для серверних систем. Проведено огляд методів прогнозування обчислювальних ресурсів, таких як авторегресійних моделей та нейроних мереж, а саме авторегресійний метод, авторегресійне ковзного середнього, авторегресійне інтегроване ковзне середнє та метод групового урахування аргументів.

Аналіз результатів прогнозування методами AR, ARMA, ARIMA, GMDH, показав, що при прогнозуванні потреби ресурсів, кожен дає різний прогноз на одному і тому ж проміжку часу та з різним квадратичним середнім відхиленням. На основі виконаного порівняльного аналізу було визначено напрямок дослідження та остаточно сформульовано задачу. Був розроблений та запрограмований алгоритм на мові програмування C#, на основі авторегресійних моделей та нейронних мереж, а саме AR, ARMA, ARIMA, GMDH. Розроблений алгоритм дозволяє наближено визначити необхідні і достатні параметри для оцінки стану завантаженості ВМ.

Напрямками подальшого дослідження можуть бути:

- збільшення кількості методів прогнозування;
- поєднання програмного забезпечення прогнозування завантаженості з програмним забезпеченням управління віртуалізованими обчислювальними ресурсами серверної системи в умовах хмарного обчислення;
- збільшення кількості параметрів завантаженості ВМ, таких як оперативної пам’яті, мережі.

За напрямком дослідження було опубліковано такі матеріали:

- прогнозування потреби ресурсів на основі гібрідного алгоритму;
- порівняльний аналіз засобів моделювання інфраструктури хмарних обчислень;
- динамічне розміщення віртуальних машин на основі навчання з підкріпленням в хмарних центрах обробки даних[6];

Отже, результати, які отримані при дослідженні роботи прогнозування потреби ресурсів для серверної системи, показали кращу точність прогнозування завантаженості CPU.

ПЕРЕЛІК ПОСИЛАНЬ

1. Толстікова О.В. Завдання створення центру обробки даних великого виробництва [Текст]: О.В. Толстікова, В.І. Дрововозов. – Проблеми інформатизації та управління., 2015 – С.123-128.
2. N. Kamiyama, "Designing Data Center Networks Using Analytic Hierarchy Process," 2010 Proceedings of 19th International Conference on Computer Communications and Networks, Zurich, 2010, pp. 1-6.
3. Терентьєв Р.А., Жаріков Е.В. Порівняльний аналіз засобів моделювання інфраструктури хмарних обчислень / Р.А. Терентьєв, Е.В. Жаріков / Матеріали 10-ї Всеукраїнської науково-практичної Web конференції аспірантів, студентів та молодих вчених «Комп'ютерні інтелектуальні системи та мережі». – м. Кривий-Ріг.: ДВНЗ «Криворізький національний університет», 22-24 березня 2017 р. – С. 8-10.
4. Жаріков Е.В., Терентьєв Р.А.. Прогнозування потреби ресурів серверної системи в умовах хмарних обчислень / Е.В. Жаріков, Р.А. Терентьєв / Матеріали науково-практичної конференції «Інформатика та обчислювальна техніка-IOT-2018». – м. Київ.: НТУУ «КПІ ім. Ігоря Сікорського», 23-24 квітня 2018 р.
5. Science and technology of the XXI century: Section 4: Modern Information Technologies (A-L), December 07, 2017 [Електронний ресурс] // Режим доступу: http://konfist.fl.kpi.ua/sites/default/files/section_4_modern_information_technologies_a-l.pdf.
6. Жаріков Е.В. Динамічне розміщення віртуальних машин на основі навчання з підкріпленням в хмарних центрах обробки даних / Е.В. Жаріков, А.А. Коваль, Р.А. Терентьєв. // Наукові вісті Далівського університету. - 2017. - № 13.
7. Терентьєв Р. А. Вплив розміру тренувальних даних на точність прогнозу потреби ресурсів серверних систем в умовах хмарних обчислень / Р.А. Терентьєв. // Актуальные научные исследования в современном мире, 2018. – С. 107-116

8. Вичужанін В.В. Розвиток інфраструктурних рішень для технології хмарних обчислень [Текст]: /В.В. Вичужанін // Одеського національного морського університету., 2013 – С.135-148.
9. Матвєєв І.М. Віртуалізація Обчислень І Економічні Показники Корпоративної Іт Інфраструктури [Текст]: /І.М. Матвєєв //Вісник Бердянського університету менеджменту і бізнесу., 2015 – С.60-63.
10. Стіренко С.Г. Підвищення ефективності роботи ІТ інфраструктури на основі технології віртуалізації [Текст]: /С.Г. Стіренко, Д.О. Шаурін. //Вісник НТУУ «КПІ» Інформатика, управління та обчислювальна техніка №49. – С.128 – 134.
11. Теленик С.Ф. Генетичні алгоритми вирішення задач управління ресурсами і навантаженням центрів оброблення даних [Текст]: /С.Ф. Теленик, О.І. Ролік, М.М. Букасов, С.А. Андросов. – В надзаг.: Інформаційно-управляючі комплекси і системи. – С.106 – 120.
12. Бідюк П.І. Системний підхід до прогнозування на основі моделей часових рядів [Текст]: /П.І. Бідюк. – Системні дослідження та інформаційні технології №3. - С.88 — 110.
13. Гузій М. М. Аналіз технологій моніторингу комп’ютерних мереж [Текст]: /М. М. Гузій, О. В. Станіславова, М. В. Кадет. – В надзаг. – Наукоємні технології №1. - С.46 — 50.
14. Теленик С.Ф. Технологія управління ІТ-інфраструктурою на основі ресурсного підходу [Текст]: /С.Ф. Теленик, О.І. Ролік, М.М. Букасов// Вісник ЖДТУ. – № 4 (47) .– В надзаг.: Технічні науки. – С.180 – 189.
15. Дзендзелюк О. Побудова ARIMA моделей часових рядів для прогнозування метеоданих на мові програмування [Текст]: /О. Дзендзелюк, Л. Костів, В. Рабик. //Електроніка та інформаційні технології №3. - С.211 — 219.
16. Скатков А.В. Информационная модель управления ИТ-ресурсами критических систем [Текст]: /А.В. Скатков, К.П. Аникевич, В.И. Шевченко.– В надзаг.: Інформаційні системи і технології – С.201 – 206.
17. Стіренко С.Г. Ефективне застосування технології віртуалізації для підвищення роботи ІТ інфраструктури [Текст]: /С.Г. Стіренко, Ю.А. Тимошин. //Матеріали

конференції «Проблеми інформатизації та управління»: збірка наукових праць – С.125 – 130.

18. Beloglazov A. Energy Efficient Resource Management in Virtualized Cloud Data Centers [Текст]: матеріали of 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, The University of Melbourne, Australia, 2010: тези / Anton Beloglazov, Rajkumar Buyya. – С.826 – 831.
19. A. A. Bankole and S. A. Ajila, "Predicting cloud resource provisioning using machine learning techniques," 2013 26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Regina, SK, 2013, pp. 1-4.
20. R. Hu, J. Jiang, G. Liu and L. Wang, "KSwSVR: A New Load Forecasting Method for Efficient Resources Provisioning in Cloud," 2013 IEEE International Conference on Services Computing, Santa Clara, CA, 2013, pp. 120-127.
21. M. Dhingra, J. Lakshmi, S. K. Nandy, C. Bhattacharyya and K. Gopinath, "Elastic Resources Framework in IaaS, Preserving Performance SLAs" 2013 IEEE Sixth International Conference on Cloud Computing, Santa Clara, CA, 2013, pp. 430-437.
22. J. Fito, I. Goiri, and J. Guitart, “Sla-driven elastic cloud hosting provider,” in Parallel, Distributed and Network-Based Processing (PDP), 2010 18th Euromicro International Conference on, feb. 2010, pp. 111 –118.
23. C. Jiang, X. Xu, J. Zhang, Y. Li, and J. Wan, “Resource allocation in contending virtualized environments through vm performance modeling and feedback,” in Chinagrid Conference (ChinaGrid), 2011 Sixth Annual, 2011, pp. 196–203.
24. Y. Jiang, C. shing Perng, T. Li, and R. Chang, “Self-adaptive cloud capacity planning,” in Services Computing (SCC), 2012 IEEE Ninth International Conference on, 2012, pp. 73–80.
25. Z. Gong, X. Gu, and J. Wilkes, “Press: Predictive elastic resource scaling for cloud systems,” in Network and Service Management (CNSM), 2010 International Conference on, oct. 2010, pp. 9 –16.
26. Z. Shen, S. Subbiah, X. Gu, and J. Wilkes, “Cloudscale: elastic resource scaling for multi-tenant cloud systems,” in Proceedings of the 2nd ACM Symposium on Cloud Computing, ser. SOCC ’11. New York, NY, USA: ACM, 2011, pp. 5:1–5:14.

27. M. Dhingra, J. Lakshmi, and S. K. Nandy, “Resource usage monitoring in clouds,” in Grid Computing (GRID), 2012 ACM/IEEE 13th International Conference on, sept. 2012, pp. 184 –191.
28. Часові ряди. [Електронний ресурс] // Режим доступу: http://www.pu.if.ua/depart/Statistics/resource/file/%D0%A7%D0%B0%D1%81%D0%BE%D0%B2%D1%96%20%D1%80%D1%8F%D0%B4%D0%B8.pdf.
29. Лук’яненко І. Г. Аналіз часових рядів побудова arima [Текст]: /І. Г. Лук’яненко, В. М. Жук // Вісник "Києво-Могилянська Академія", 2013.- С. 6-184
30. Афанасьев В.Н. Аналіз часових рядів і прогнозування [Текст]: / В.Н. Афанасьев, М.М. Юзбашев // Финансы и статистика.html.
31. Регресійний аналіз # [Електронний ресурс] // Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Регрессионный_анализ/.
32. Авторегресія (AR, autoregression) # [Електронний ресурс] // Режим доступу: https://fnow.ru/articles/avtoregressia.
33. Авторегресійне ковзне середнє # [Електронний ресурс] // Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Авторегрессионное_скользящее_среднее.
34. Arima # [Електронний ресурс] // Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Autoregressive_Integrated_Moving_Average
35. Нейронні мережі, сутність мереж # [Електронний ресурс] //Режим доступу: http://opticstoday.com/katalog-statej/stati-na-ukrainskom/nejromerezhi/nejronni-merezhi-sutnist-merezh.html
36. Мак-Каллок У. С. Логічне числення ідей, що відносяться до нервової активності [Текст]: У. С. Мак-Каллок, В. Піттс. // «Автомати» під ред. К. Е. Шеннона і Дж. Маккарті. - М, 1956. – С.363-384
37. Бодянский Е.В. Радіально-базисна нейронна мережа з поліноміальних функциями активації [Текст]: Е.В. Бодянский, А.П. Чапланов, Е.Б. Чапланова. // Системи обробки інформації, 2007. – С.12-16

38. Івахненко А.Г. Індуктивний метод самоорганізації моделей складних систем [Текст]: /А.Г. Івахненко // Наукова думка, 1981.
39. Івахненко А.Г. Перешкодостійкість моделювання .[Текст]: /А.Г. Івахненко, В.С. Степашків. //Наукова думка, 1985.
40. Методи індуктивного породження регресійних моделей [Електронний ресурс] // Режим доступу: http://www.machinelearning.ru/wiki/images/b/b9/Strijov08ln.pdf
41. Метод групового урахування аргументів [Електронний ресурс] // Режим доступу:http://machinelearning.ru/wiki/index.php?title=Метод_группового_учета_аргументов
42. Group Method of Data Handling (GMDH) for deep learning, data mining algorithms optimization, fuzzy models analysis, forecasting neural networks and modeling software systems [Електронний ресурс] // Режим доступу: http://www.gmdh.net/
43. J. Xiao, Y. Hu and S. Wang, "Complex-Valued GMDH-type Neural Network for Real-Valued Classification Problems," 2013 Sixth International Conference on Business Intelligence and Financial Engineering, Hangzhou, 2013, pp. 70-74.
44. Основні принципи і загальна схема МГУА. [Електронний ресурс] // Режим доступу: http://iasa.org.ua/lections/tpr/mgua/general.htm
45. Мороз А.Г. Переборний алгоритм МГУА з генетичним пошуком оптимальної моделі [Текст]: А.Г. Мороз. //Керуючі системи та машини, 2016. – С. 73-79.
46. Штучний інтелект Алгоритм МГУА [Електронний ресурс] // Режим доступу: http://www.ievbras.ru/ecostat/Kiril/Library/Content393/Content393
47. Багаторядні поліноміальний алгоритми МГУА [Електронний ресурс] // Режим доступу: http://iasa.org.ua/lections/tpr/mgua/polinom.htm
48. C#: Використання середовища розробки Visual C # [Електронний ресурс] // Режим доступу: https://msdn.microsoft.com/.
49. Visual Studio: Повнофункціональна інтегроване середовище розробки (IDE) для Android, iOS, Windows, хмари і Інтернету [Електронний ресурс] // Режим доступу: https://www.visualstudio.com/ru/vs/.
50. R: The R Project for Statistical Computing [Електронний ресурс] // Режим доступу: https://www.r-project.org/.

51. Що таке NuGet і для чого він потрібен | Microsoft Docs [Електронний ресурс] // Режим доступу: https://docs.microsoft.com/uk-ua/nuget/what-is-nuget.
52. GWA-T-12 Bitbrains [Електронний ресурс] // Режим доступу: http://gwa.ewi.tudelft.nl/fileadmin/pds/trace-archives/grid-workloads-archive/datasets/gwa-t-12/fastStorage.zip

ДОДАТОК А Графічний матеріал

ПЛАКАТ 1 Блок-схема алгоритму динамічного розміщення віртуальних машин на основі навчання з підкріпленням

ПЛАКАТ 2 Математична модель

ПЛАКАТ 3 Структура вхідних і вихідних даних

ПЛАКАТ 4 Схема структурна класів програмного забезпечення

ПЛАКАТ 5 Копії екранних форм

ПЛАКАТ 6 Результати досліджень (1)

ПЛАКАТ 7 Результати досліджень (2)